



Fall 2019 Individual Report FAS-COMPSCI 121-Introduction to Theoretical Computer Science 001 Boaz Barak

Project Title: **2019 Fall Harvard FAS Course Evaluation**

Course Audience: **185**

Responses Received: **159**

Response Ratio: **86%**

Report Comments

Note:

The order that the questions appear on this report is not the same as the way the questions were displayed to students. The order has been changed to make the report more readable.

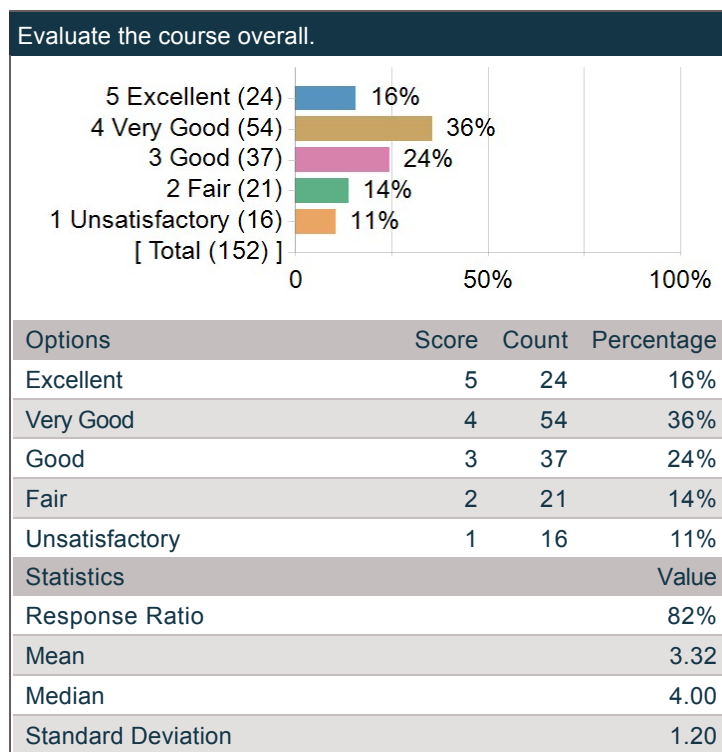
Creation Date: **Saturday, January 11, 2020**

General Course Questions

Course General Questions

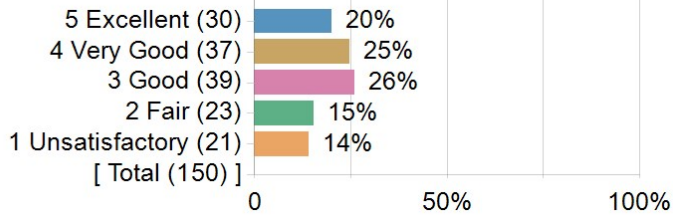
	Count	Excellent	Very Good	Good	Fair	Unsatisfactory	Course Mean	FAS Mean
Evaluate the course overall.	152	16%	36%	24%	14%	11%	3.32	4.17
Course materials (readings, audio-visual materials, textbooks, lab manuals, website, etc.)	150	20%	25%	26%	15%	14%	3.21	4.15
Assignments (exams, essays, problem sets, language homework, etc.)	149	16%	28%	24%	21%	10%	3.19	4.00
Feedback you received on work you produced in this course	149	13%	24%	28%	25%	10%	3.06	3.95
Section component of the course	99	25%	26%	30%	12%	6%	3.53	4.10

Evaluate the course overall.



Course materials (readings, audio-visual materials, textbooks, lab manuals, website, etc.)

Course materials (readings, audio-visual materials, textbooks, lab manuals, website, etc.)



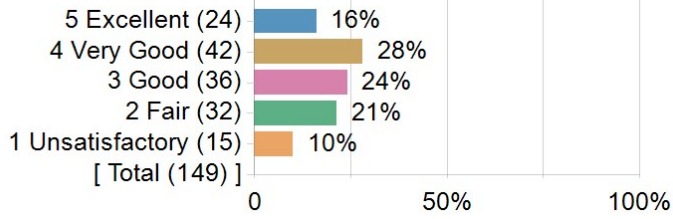
Options	Score	Count	Percentage
Excellent	5	30	20%
Very Good	4	37	25%
Good	3	39	26%
Fair	2	23	15%
Unsatisfactory	1	21	14%
Statistics	Value		
Response Ratio	81%		
Mean	3.21		
Median	3.00		
Standard Deviation	1.31		

Course materials (readings, audio-visual materials, textbooks, lab manuals, website, etc.) (Comments)

Comment
- Some typos in the textbook
- The book is a work in progress, is excessively verbose, and unhelpful in most respects.
- Great book!
- Boaz continues to ignore typos in the textbook
- Textbook has many errors.
- Book is solid. For some reason the class had 4 websites (cs121, canvas, gradescope, ed). which was confusing
- Really appreciate the effort that has gone into making the textbook!
- One of the best cs text books ive used
- The textbook is difficult to read/understand and the covered a lot more than we ever spoke about in class.
- The textbook is in the works and as such is riddled with typos. there are also not practice problems with old answers posted and the section notes / answers were posted way after leading to 0 ways to practice the material besides for the problem sets.
- Textbook is hard to read, incomplete, and has many typos.
- I like the textbook, even though a lot of people don't.
- There are some typos in the book still.
- Please finish the textbook :(
- The textbook at times contained inconsistencies and errors.
- The online textbook worked very poorly with Chrome.
- lectures are tough and the textbook requires a lot of reads to be helpful
- the book is a work in progress though one day I think it will be excellent
- Textbook is atrocious
- The textbook could be hard to understand at times
- Textbook is decent but needs work.
- My preferred mode of learning was reading the textbook
- I mostly learned the course content out of the textbook, which I thought was well done. The reading guidelines on the course website were invaluable!
- some parts of thrme textbook were completely missing
- Textbook still needs to be improved, some proofs were way too complicated and difficult to understand and it's unfinished
- some textbook chapters missing?
- Textbook was confusing in places and had multiple typos.
- Website was great, textbook is still coming together.
- Good textbook
- The textbook needs some rigorous proofreading.
- Free and useful
- The textbook is shamefully bad. Boaz has conscripted an entire class of CS concentrators to edit his textbook while they attempt to learn from it, which makes no sense. Pick a textbook that is a) already a finished product and b) explains concepts in an approachable way. Writing textbooks is not for everyone, clearly.
- Though I didn't find the lecture slides that helpful, the textbook was extremely detailed and comprehensive.

Assignments (exams, essays, problem sets, language homework, etc.)

Assignments (exams, essays, problem sets, language homework, etc.)



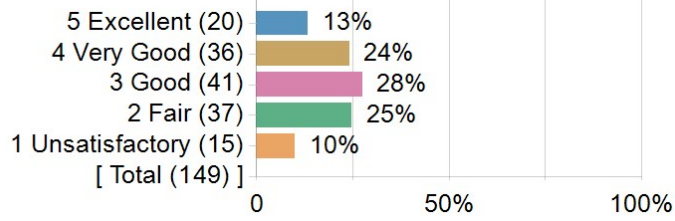
Options	Score	Count	Percentage
Excellent	5	24	16%
Very Good	4	42	28%
Good	3	36	24%
Fair	2	32	21%
Unsatisfactory	1	15	10%
Statistics			Value
Response Ratio			81%
Mean			3.19
Median			3.00
Standard Deviation			1.23

Assignments (exams, essays, problem sets, language homework, etc.) (Comments)

Comment
- Second midterm was difficult
- One midterm was way too long, problem sets are exceedingly wordy and unduly stressful.
- I appreciated the bonus points, though sometimes the questions felt tricky in a way that didn't really test our understanding.
- Psets mostly fine, sometimes the problems can have too many boring details that lose the theoretically interesting bits. Exams were great except that they forced us to memorize TONS OF BORING TERMS instead of just providing the definitions on the exam & testing how we can use them.
- Assignments usually feel motivated. Quizzes are fine, but communication was unclear.
- there are a lot of psets where I honestly should have just written "I don't know" than giving an honest attempt; it felt like I was wasting my time
- The exams were arbitrary and focused on particular topics that did not reflect all of the material – having so few questions made the exams hit or miss.
- Psets have too many bonus points, exams require too much parsing of notation and are unintuitive.
- The problem sets were very long for the amount of time given to solve them, and many of the problems were unreasonable in difficulty. The lectures were ineffective for understanding some of the material, which further exacerbated the stress and confusion caused by the problem sets and exams.
- WAY TOO DIFFICULT
- The psets could have been a little more challenging.
- really long
- Psets could be condensed considerably to not be "busy work"
- Really difficult at times (certain problem sets like 5 more difficult than others). Approaches to problems unclear until after completing them
- The difficulty of the psets and exams were inconsistent, but after most people scored low second midterm, they did respond with adjusting the final.
- Exams were difficult but fair. Problem sets were unreasonably difficult and impossible to do without course-sanctioned aid.
- The homework teaches you a lot, though is sometimes a bit overwhelming, as it should be.
- The homeworks had a large gap in difficulty between what we learned and what we needed to do. However, it prepared you well for exams. The exams were hit and miss, as #1 and #3 were good but #2 was way too long
- Exams were fine– homework ranged a lot in quality.
- The second midterm was hard, but the first midterm and the final felt reasonable
- The problems in the homework were often a little convoluted. This may have been intentional, but it usually took me longer to read and understand what the problems were asking than to actually apply concepts from class to solve them. This made the homework not very enjoyable.
- Some assignments were unnecessarily tedious and the second midterm was not written well!
- Second to last problem set was pretty much impossible for everyone to do without TF's pretty much walking through it in office hours.
- There were some really interesting problems, but the difficulty and length of assignments varied wildly.
- Difficult but doable, and lots of bonus points available
- Never too hard and usually interesting
- The PSETs were well-designed in that they ensured understanding of all the important concepts without being redundant. After doing the PSETs, I was able to go back and actually understand much of the material in the textbook. Additionally, the inclusion of many bonus points made PSETs and exams much less stressful.

Feedback you received on work you produced in this course

Feedback you received on work you produced in this course

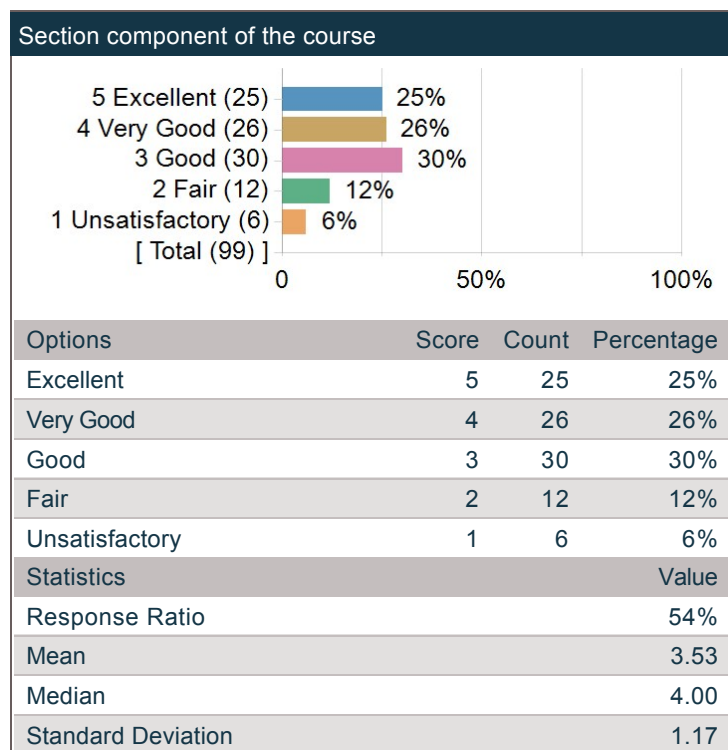


Options	Score	Count	Percentage
Excellent	5	20	13%
Very Good	4	36	24%
Good	3	41	28%
Fair	2	37	25%
Unsatisfactory	1	15	10%
Statistics			Value
Response Ratio			81%
Mean			3.06
Median			3.00
Standard Deviation			1.20

Feedback you received on work you produced in this course (Comments)

Comment
- Large discrepancy in grading across TFs.
- Although I (superficially) understood the staff solutions, I struggled to understand why my answer was incorrect.
- I do wish they were graded faster.
- Overused rubric, sometimes would have over specific points like "used Shannon's Entropy Theorem, -10" which like ... I get why I lost points, per the rubric, but this is not helpful since I can't see the rubric and it wasn't at all clear how these rubrics are devised. Rubrics are only really helpful if released ahead of time!
- TF's really don't seem to understand the material well enough to be either teaching or grading it.
- Had to frequently request for regrades (and for good reason)
- Marking pretty poor at times
- Some of the TFs did not seem to really understand the problem sets, I ended up requesting multiple regrades on problems where the TF's feedback was wrong (and how helpful OHs were also depended extremely on whether there was a competent TF around).
- Grades, and comments came back very late except for the exams and as such it was tough to incorporate them or your understanding into studying for the tests or future problem sets.
- Had to ask for regrades quite often
- Psets had minimal explanations of incorrect answers
- some graders are ridiculous
- CAs require WAY more consistency
- Sometimes rather untimely
- Sometimes got docked points with no explanation
- The grading was very inconsistent and it really felt like some TFs were really strict, while others were a little nicer
- Odd timing– where much of the feedback for the early psets comes after the first midterm so it's not very useful
- Some TFs gave great feedback, and some gave poor feedback/had consistently incorrect counter–examples
- TF's did a good job of grading.
- I would have liked more feedback on exams
- The Gradescope system works extremely well and teaching staff was always very quick to grade and respond to regrade requests.
- Pretty timely feedback, not a fan of how the assignments were graded (taking points away instead of giving points)
- Sometimes feedback was unclear and points were deducted for unclear reasons ("point adjustment")
- Feedback was limited and not returned in a timely manner.
- Sometimes the comments were a bit vague, but the timeliness of grading was great!
- Slow grading, seemed liked TF's barely looked at psets
- Could've used quicker and more in–depth feedback
- grading seemed to be not super consistent
- I thought some of the grading criteria were unnecessarily harsh.

Section component of the course



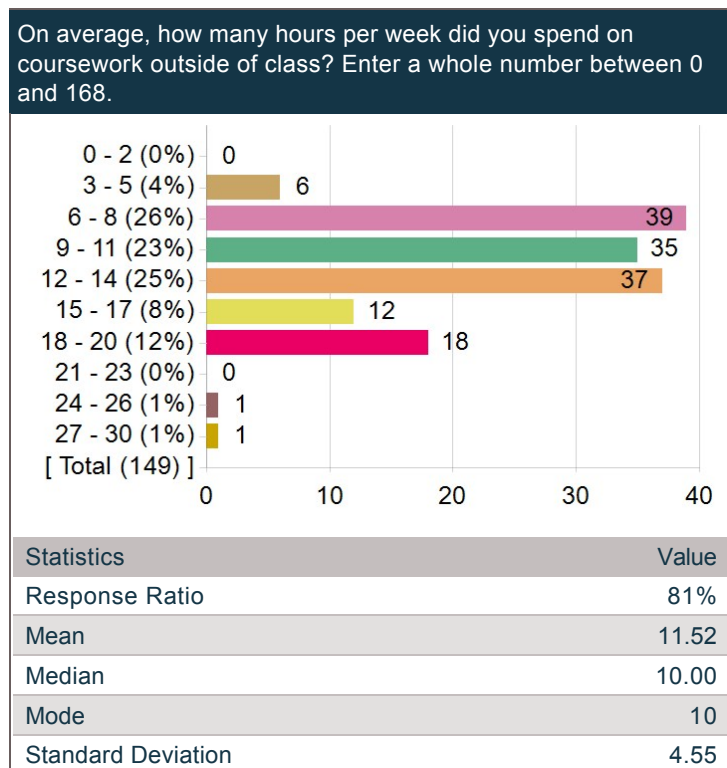
Section component of the course (Comments)

Comment
- I only attended the advanced section semi regularly but it was great.
- i don't know anyone that attended and the section notes were posted only before the exams.
- Sections, like lecture, are just reviewing definitions
- Go to section!
- I was on available for 2 slots and both TFs struggled to answer questions
- Will's section was incredible! I found it early and kept telling people to come because it was that good.
- Did not attend
- Section is a good way to reinforce material
- Probably the best portion of the course
- I didn't attend any sections.
- very useful in breaking down the abstract concepts of the class
- 121.5 was great!
- Generally could be better focus on what is IMPORTANT in the course
- I only attended the advanced section
- Some sections were really helpful, others are a waste of time.
- Sections were truly a turning point for this class (especially Will B.'s)
- Did not attend
- I would have preferred to be assigned to a section then I might have actually gone

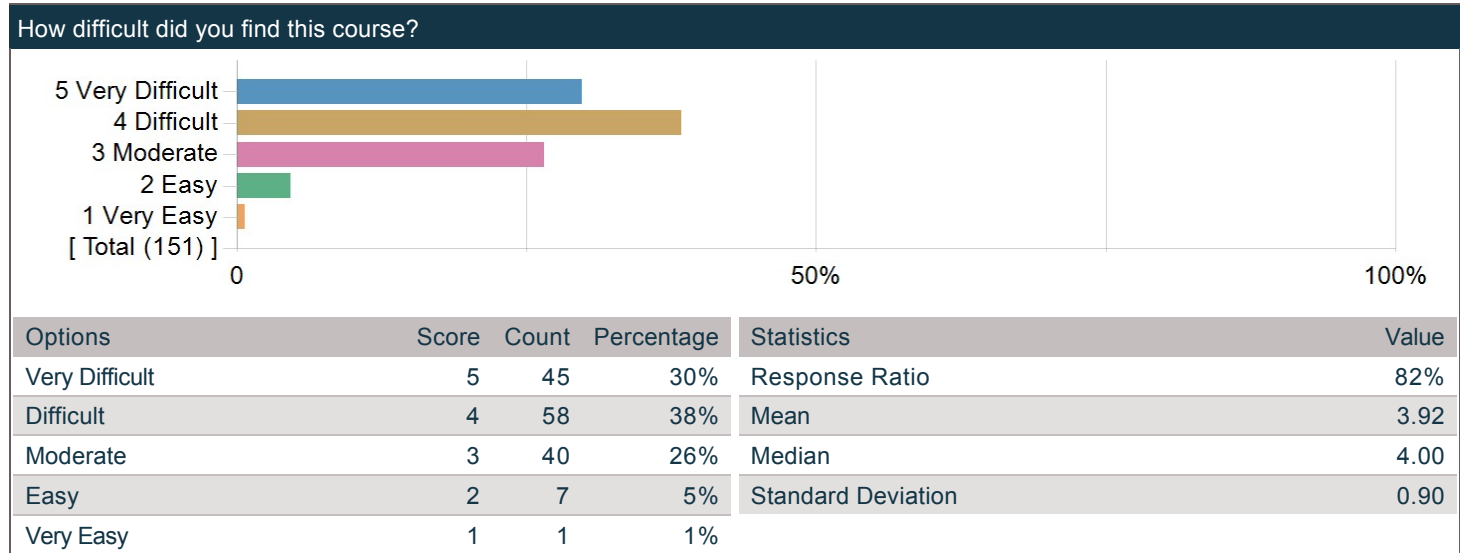
Requirements - What did this course require of you?

On average, how many hours per week did you spend on coursework outside of class? Enter a whole number between 0 and 168.

The top range is 168 but displayed below are frequency bars for those who answered up to 30. The mean reflects the entire range.



How difficult did you find this course?

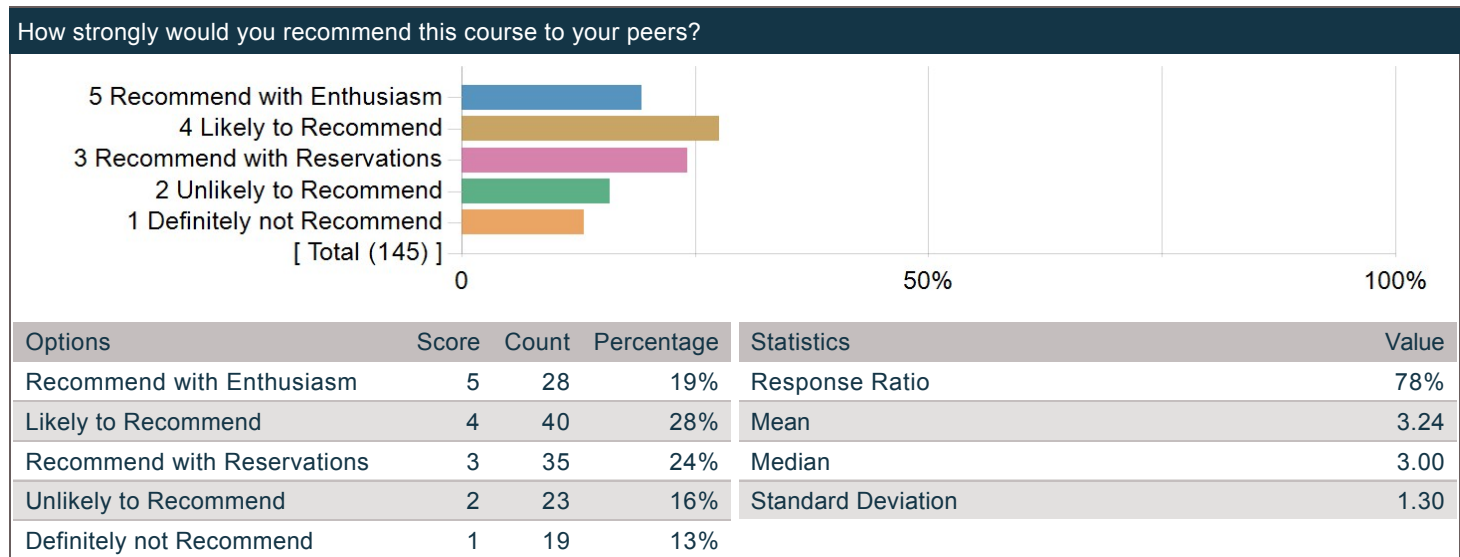


What was/were your reason(s) for enrolling in this course? (Please check all that apply)

Options	Count
Elective	12
Concentration or Department Requirement	142
Secondary Field or Language Citation Requirement	7
Expository Writing Requirement	1
Divisional Distribution Requirement	1

Recommendations - Would you recommend this course?

How strongly would you recommend this course to your peers?

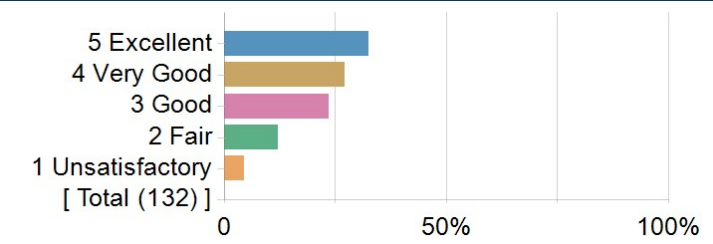
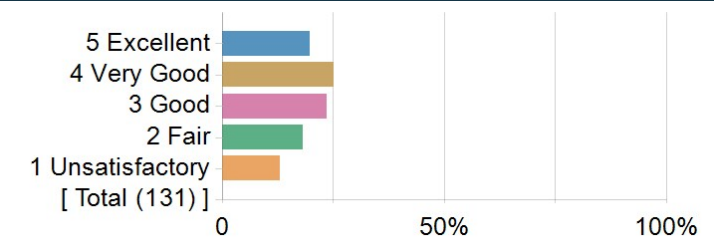


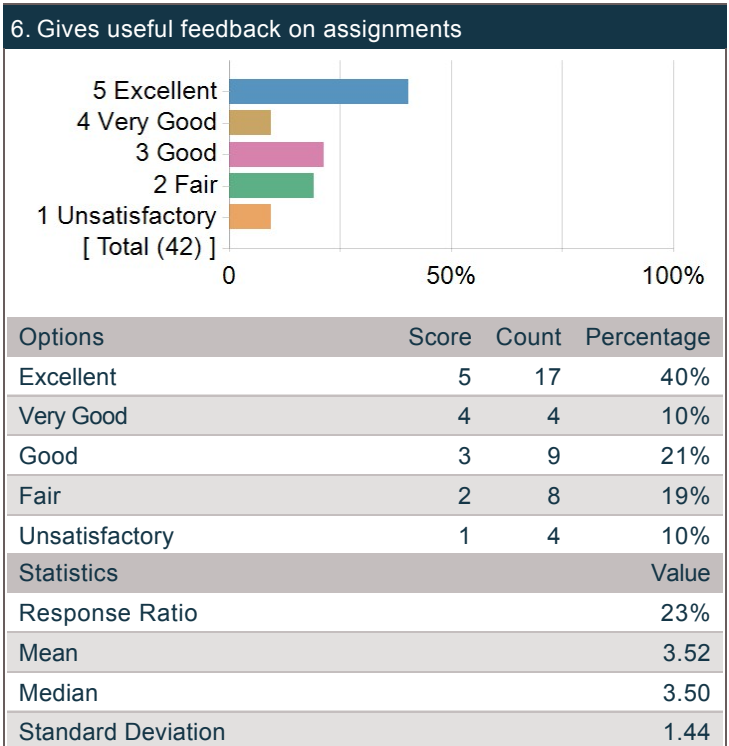
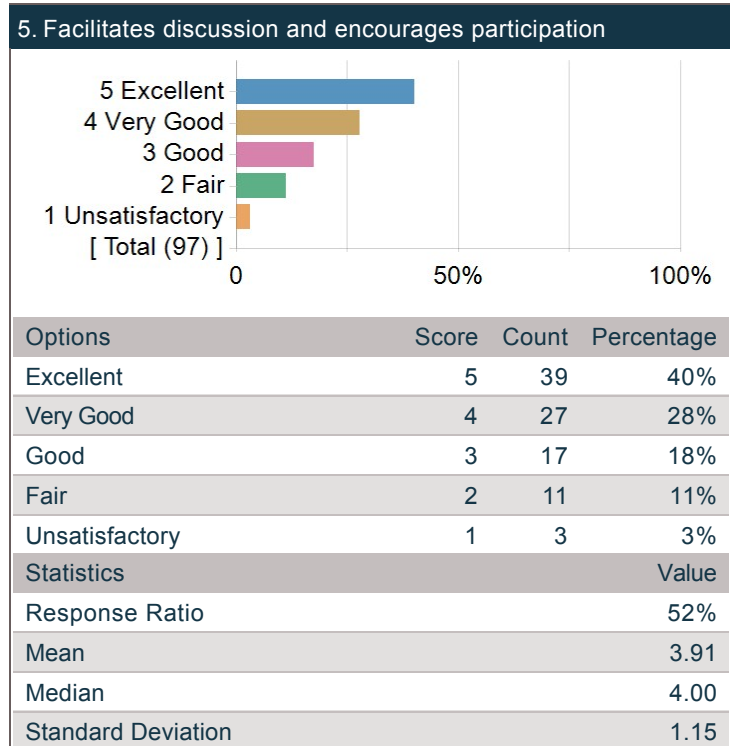
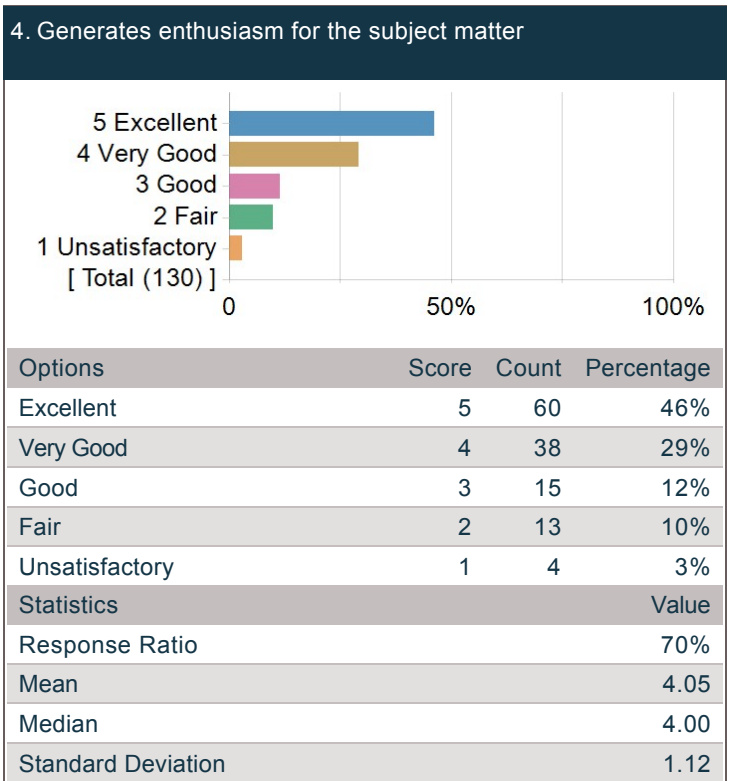
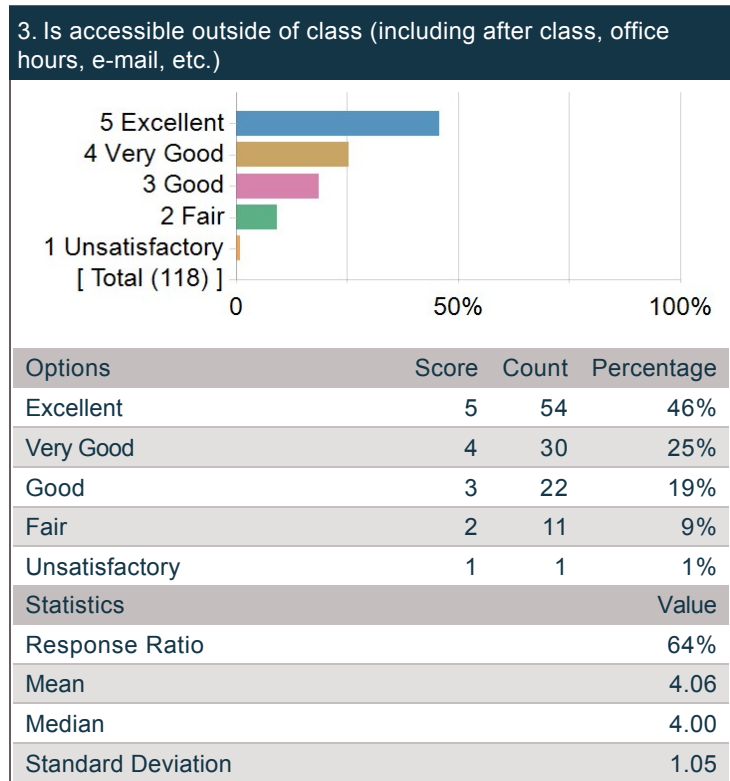
Evaluation of Instructors

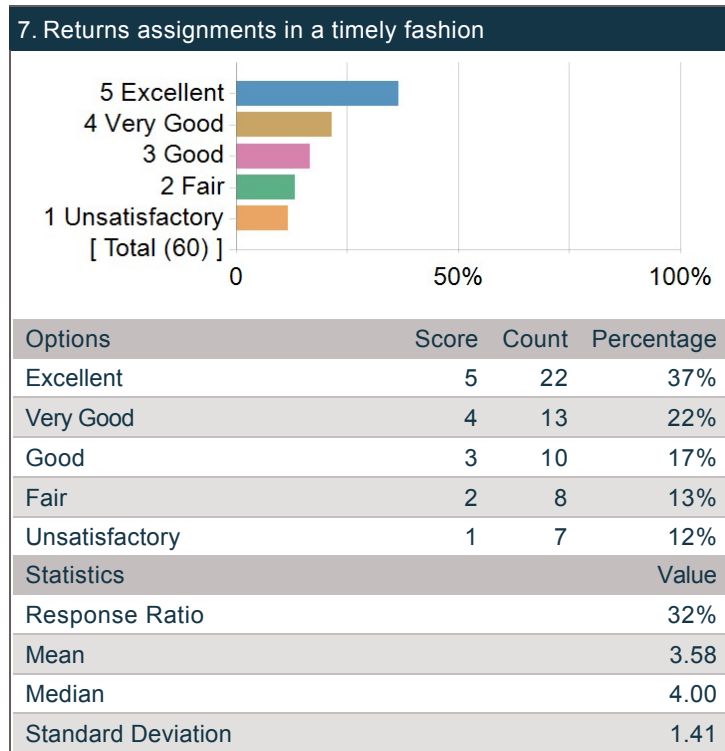
General Instructor Questions

	Count	Excellent	Very Good	Good	Fair	Unsatisfactory	Instructor Mean	FAS Mean
Evaluate your Instructor overall.	132	33%	27%	23%	12%	5%	3.71	4.48
Gives effective lectures or presentations, if applicable	131	20%	25%	24%	18%	13%	3.21	4.36
Is accessible outside of class (including after class, office hours, e-mail, etc.)	118	46%	25%	19%	9%	1%	4.06	4.35
Generates enthusiasm for the subject matter	130	46%	29%	12%	10%	3%	4.05	4.52
Facilitates discussion and encourages participation	97	40%	28%	18%	11%	3%	3.91	4.42
Gives useful feedback on assignments	42	40%	10%	21%	19%	10%	3.52	4.35
Returns assignments in a timely fashion	60	37%	22%	17%	13%	12%	3.58	4.39

Instructor

1. Evaluate your Instructor overall.					2. Gives effective lectures or presentations, if applicable				
									
Options	Score	Count	Percentage		Options	Score	Count	Percentage	
Excellent	5	43	33%		Excellent	5	26	20%	
Very Good	4	36	27%		Very Good	4	33	25%	
Good	3	31	23%		Good	3	31	24%	
Fair	2	16	12%		Fair	2	24	18%	
Unsatisfactory	1	6	5%		Unsatisfactory	1	17	13%	
Statistics				Value	Statistics				Value
Response Ratio				71%	Response Ratio				71%
Mean				3.71	Mean				3.21
Median				4.00	Median				3.00
Standard Deviation				1.18	Standard Deviation				1.31





General Course Questions - Comments

What were the strengths of this course? Please be specific and use concrete examples where possible.

Comments
The material was so amazing! And so was Boaz Barak!!! He cared so so so much about the class and about us and it felt like a very nice intellectual community.
Good content. Organized course.
Sections and office hours are absolutely important in understanding concepts. Concepts are well-taught and explained in both of these resources. The earlier parts of the course were very well taught, especially uncomputability and rice's theorem.
Good office hours
It's mandatory.
The overall content was great and the lecture was good
Boaz clearly cared a lot about the course and put a lot of energy into making sure that we had support for the course.
It was well paced and well taught.
What I really really like about this course is that it is the only course I've taken at Harvard that had given me a glimpse of the "edge" of human knowledge. After the first midterm, we constantly encounter conjectures that we have no idea how to prove — and man that gives me an intellectual orgasm! Usually in a course you learn what people have figured out 100+ years ago, there's a bit of that in the course, but you also learn what people have yet to figure out for the 100+ years ahead, and that is exciting!
Also, the online support forum is superb. Boaz and all the TF actually cares about us learning — really, learning about critical thinking, asking good questions and understanding complex propositions — not in the sense of solving the pset at hand. That is something to applaud to!
The problem sets did help to reinforce the ideas learned in the course and the sections did a good job in putting the concepts in more understandable terms.
The content was really interesting, especially towards the end as we got into efficient and randomized computation! The teaching staff, especially CAs and TFs were always very willing to help and make sure we understood.
CS121 does a good job covering important topics in theoretical CS. It seems like the course material has been carefully thought out to create a modern approach to the field which prominently features unsolved problems and open research questions. It is also clear that Boaz and the teaching staff have made a considerable effort to reduce student anxiety about this course.
Feedback on assignments was thorough and prompt, and we were given a chance to get clarification on feedback.

Comments
Some interesting content.
The main strength was in the efforts of the teaching staff. Boaz was clearly trying to make this course as good as it could be, and all of the TFs were devoted to helping the students. Though I have complaints about the course material, I have no complaints about the people who taught it; and with their clear effort to improve the course and how much it's changed even recently, I have confidence they'll make the course into something worth taking.
The optional advanced section is really cool and introduces a lot of interesting topics. The teacher is very knowledgeable and willing to help the students. The subject material was very interesting. He recommends a lot of books relating to the course material and I really appreciated that.
Taught big ideas and concepts quite well. I understand intuitively what it means for a computational system to be equivalent to a Turing Machine / circuit / series of circuits, or to be uncomputable.
Professor is very dedicated to the class and really cares for his students – very receptive to feedback and has a good sense of humor Lots of resources in place for all students, e.g. CS121.5 and optional assignments for stronger students and assistance for weaker students
Taught very well and many opportunities to show what you've learned, makeup lost points.
Really interesting content
Boaz is nice
Good theoretical overview of computability and computational time.
It is an interesting course (the material is good)
–Boaz answers literally every Ed post
The open–source textbook is nice.
Boaz is a good person
Very comprehensive material, great support system for students
This course provided a lot of support to its students, and I appreciated the genuine effort and care that went into helping students achieve success. It was obvious that a lot of investment went into the course.
The textbook, lectures, and section materials covered the content very comprehensively. Lectures were great for content overview and big picture intuition, the textbook was helpful for understanding proof and theorem details, and sections helped explain problem–solving techniques and rephrased the material in a more accessible way. TFs and Boaz were always super prompt in responding to questions on Ed and seemed enthusiastic about the course and about helping us learn.
Amazing teaching staff, riveting material.
The material is actually very interesting and Boaz + the staff clearly put in lots of effort. The better among the problem set questions are really illuminating and help you learn a lot.
The course marking scheme was designed very well to reduce stress and improve learning outcomes. Also, the professor was phenomenal at breaking down complex topics and appeared genuinely invested in his student's learning and success.
The topics were interesting, all the things we covered are things that I want to learn about.
The psets could be a lot of fun, if you started early enough and weren't stressed.
The course definitely touches upon subjects that are needed in other fields of work.
A good treatment of mathematics as necessary for computer science
The homework problems were structured well such that they weren't too hard but you would still learn the content. Also, the advance section setup was good.
TFs are great, material is interesting
The material was fascinating. Ed was very active and helpful
This course provides an excellent introduction to the theory of computation, and Boaz found a great way to teach the course in a way where everything connects and makes sense together. I've often felt that I really don't like theoretical computer science and have always struggled in CS courses that are very theoretical (as most are at Harvard), however this course was taught in such a way that I have a much better appreciation for theoretical CS and could even see myself trying to continue my education in CS theory after graduation. I thought I was going to hate this course, but I often found that I enjoyed solving the problems more than I had anticipated. Although this class was challenging, it was also extremely rewarding. Boaz clearly cares about every student succeeding and understanding the material, and I really appreciate the work he put into this course, as well as the TFs. I wish I had taken this course earlier in my college career, partially because it would have been very helpful for CS 124, and partially because I probably would have pursued more theoretical courses had this been my first real introduction to CS theory courses.
Content is really interesting, TFs are great
I think that the primary strength of the course was the recognition on the part of course staff that the material is difficult.

Comments
Boaz is a really sweet guy and very responsive and willing to help. Course staff is active on Ed. TF William Burke is a LIFESAVER. The weighting of the midterms vs psets vs finals were good, and I appreciated the ample extra credit opportunities.
Lots of support from professors and teaching staff, adjusted according to feedback, a thorough textbook that directly related to lectures, higher level lectures that are helpful for overall understanding
This course taught me what I wanted to learn in a way that interested me.
The subject material is interesting, the expectations are very well specified. The bonus point system is really great.
A lot of difficult material was covered efficiently, and lectures were usually very intriguing. Additionally, the TFs especially put in a lot of work to ensure that students understood the content presented.
All the material works together and one might not see it at first, but you really appreciate the system of knowledge developed by the end of the course.
Interesting Material
Professor Boaz is incredibly kind in office hours.
<ul style="list-style-type: none"> – you cover a lot of material – textbook is somewhat helpful – TFs are available frequently – Boaz cares about student well-being even if he doesn't really know how to deal with student stress and panic – exams are stressful and difficult but fair – the trick-or-treat reduction guide created by the TFs was really helpful
I learned how to think more mathematically. I really did improve my thinking in areas of theory. I think the class was good in teaching cool material. The lectures are usually petty good, and I feel that I can learn a lot. The book was actually not that bad once you skip over all the technical proof parts, then you can focus on the meta and learn what you need to know and actually enjoy learning. The lectures are more for filling in the gaps that the book does not cover, so overall it is well-done.
The TFs were the only redeeming portion of this course
Professor Boaz really cares about the course and students.
None
The structure of this course is very good. The progression of ideas is very logical and fits well in the grand scheme of the whole course.
Boaz gave very understandable lectures, which helped a lot in the tougher parts of the course.
lots of resources to try to help students
Interesting material, and Boaz is very friendly!
The content was really interesting and there was a lot of support available.
The material taught in this course is very engaging and interesting.
Material is interesting and course curriculum is structured well
Teaches a lot of content.
I felt that the time that I spent engaging with the problem sets was time well spent. Exams seemed to be pretty reasonable evaluations of understanding. I felt like I learned a lot from reading the textbook.
Frequency of office hours and overall support, course calendar (helped me plan out deadlines), late days, collaboration policy
Boaz is so enthusiastic! Staff is great and very committed. Material is intellectually interesting.
Boaz cares about the class, shown through his extensive extra credit and consistent reminders that the grade isn't everything. He improved the textbook and lectures significantly from last year (the textbook still has a ways to go for ease of understanding, but it'll get there in ~2 years).
Lots of support from teaching staff. Ample bonus points on problem sets to take the pressure of "perfection" off and to allow people to learn.
The professor really wants the students to succeed and he structured the class to eliminate cut-throat competition
Bonus points on many assignments, instructors were very responsive on Ed, lecture materials were readily available, general transparency about grading. Additionally, I actually appreciated that the psets were more back-loaded because it was the opposite of my other math/CS classes that have more work at the beginning of the term, so I would recommend continuing this in future years.
You get a good grasp of lots of cool algorithms and problems.
interesting material – halting, uncomputability, power of programming languages, randomized algorithms
Teaches some important concepts in theoretical computer science.

Comments
Learned a lot about theoretical CS
Material was interesting, undergraduate TFs are amazing
The course organization was very clear. We covered a diverse range of topics with context provided for the usefulness of each topic. Problem sets and exams were very fair.
Great environment, helpful staff, understanding instructors. Actively trying to improve class.
Learned a lot about how to write proofs and time complexity classes
There are so many resources available to help you stay up to speed on the course material: lectures, section, OH, textbook, etc. Tons of TFs you can talk to, all of whom lead different sections, and also additional faculty members who are just there to answer any questions you may have! Also, the problem sets were beginner-friendly and allowed students to earn points on required problems even if they didn't know where to begin.
That it ended.
Boaz really cared about the material and generally taught us the most important content. He tried his best to show us why theoretical CS matters.
Good survey to the theoretical foundations of computer science, and a helpful way to learn more about crucial concepts like runtime. Course infrastructure was clear and well-planned.
Some of the course infrastructure was really good – good exam prep resources, lots of helpful Ed posts that clarified content/additional materials like the reduction guide – some of the TFs were really helpful in going above and beyond to create good resources. I also liked the weekly quizzes – they helped make sure that students would stay caught up with the readings
This course forces you to think of things abstractly. Often times there weren't great concrete examples of the types of functions in different complexity classes, but we had to differentiate them understanding their abstract definitions.
Lectures were fairly good. I liked how many different resources are available. I also liked how much Boaz cared about the class.
<ul style="list-style-type: none"> – Incredibly responsive staff. I always felt supported. – Good and smooth class structure; it was never inconsistent (schedule is always up-to-date). Any changes were always well communicated. – Challenging and what you would expect from a Harvard course. – Great problem sets, do not induce unwarranted stress (namely because the teaching staff is so supportive.)
The Ed board was amazing and was a vital tool to learning. The fact that you could post a question about literally anything and have the professor of the course respond (usually within an hour!) is phenomenal. It also made the course feel more collaborative and accessible which was comforting.
Another strength is the proof ideas in the textbook. These were SO helpful and made understanding the proofs way easier. Especially for someone like myself with less formal math background, these were often significantly more useful than the actual proofs in the textbook.
Final strength are the bonus points. I like that there are bonus points and opportunities to make up for stupid mistakes. I found that I appreciated bonus points more towards the end of the year when they were more accessible to everyone in the course (and not those with deep math background) – especially the last problem set!
Boaz explains things very well — lectures and his posts on the discussion board (+ TFs and Madhu) were immensely helpful. PSets also reinforced the material very well.
The assessments and problem sets were extremely reasonable and Boaz was always incredibly encouraging to all his students. The grading was also very reasonable and the bonus points were quite generous.
Very interesting content which is relatively accessible
The problem sets were fun, and lecture was pretty cool. Professor Barak has a great sense of humor – he's pretty quotable when it comes to topics like sleep. Section was also the BEST ever. Thursday nights were SO good and informative.
<ul style="list-style-type: none"> –Lectures were often interesting –T-Shirts were amazing –Ed questions were answered very quickly and thoroughly –The bonus points are a great way to "curve" the class without the traditional stress of a curved class.
Interesting material but a lot of it seemed useless.
Subject material is central to theoretical computer science.
Support from the instructor and TFs. Boaz is a very nice guy and encourages us before exams! He even saw us struggling with the second midterm and gave all of us one more late day.
Furthermore, I liked how Boaz gives a high-level intuition of a proof before spitting out a lengthy CS proof in his textbook. Sometimes, the intuition is all you need to understand why a theorem works! Helps a lot especially when reviewing.

Comments
The course gives a good overview of the theoretical underpinnings of computer science, especially turing machines and complexity classes.
Could watch online from room lol.
Supportive and strong teaching staff
Problem sets were well designed and engaging. Tests were reasonable in difficulty. The textbook is super thorough and a great resource. Lots of care was put into this course.
Boaz Barak and the teaching staff are the strength of this course by far! Boaz puts in an enormous amount of effort to make sure students are comfortable and not falling behind and the teaching staff are incredibly knowledgeable, helpful, and kind.
Lectures were instructive and often entertaining, which really helped me to get enthusiastic about material that at first struck me as boring and needlessly complex.
Professor Boaz tries to make the course material very understandable, and there is a lot of opportunity for extra credit.
Good teaching, interesting content.
Lectures and the book were fantastic. 121.5 was interesting and valuable, weekly quizzes were a good addition to the course, and exams were fair.
Pset problems were approachable and exams were not terribly hard. The book is excellent as a resource and course staff was usually very responsive on ed
Lots of interesting topics covered, textbook does a good job of explaining things, a lot of knowledgeable course staff
The weekly quiz, lectures and sections together created a strong model of teaching that I had never seen put in practice with such precision and quality before. The content of the class itself is very new but so fascinating that it begs to be understood and learned.
It seems to be improving every year, and there's a lot of built-in support to help students navigate the difficult material.
Lectures (examples), course staff (friendliness and approachability).
There were lots of office hours and having many bonus problems on homeworks made it less stressful
There were no strengths of this course.
Teaches and reinforces some important computer science concepts and math problem solving techniques. Also, activities like the weekly quizzes are great for keeping everyone accountable
Lecture was great. The teaching staff was super helpful. Ed was also really helpful.
The content is well-chosen, interesting, relevant to the field of computer science, and presented in a logical order. NAND over finite automata worked well for me coming from a math background.
The course staff provided plenty of resources to try to help and genuinely cared about making sure everyone did well.
The lectures and class structure are clear and well thought out. The textbook is well written and easy to follow
You learn a lot of interesting/useful concepts in CS and a different way of thinking.
The book follows the lectures really well, so it is a great source for revision. Furthermore, the professor tries very hard to make the class as enjoyable as possible and is easily accessible outside the class.
CS 121 was a very fulfilling course, and I learned key concepts, such as computability and efficiency, as well as problem-solving strategies. Boaz cares a lot about improving the course each year, as well as emphasizing learning over grades; this is evident in the fact that there are extensive opportunities to earn bonus points (on exams and PSETs, as well as through optional projects and lecture/section attendance), and the late days make PSETs more manageable. Additionally, the textbook is extremely detailed and comprehensive. Overall, CS 121 is a very well-run course and one of the best I've taken at Harvard. Also, the complexity reduction guide was incredible and helped me learn problem-solving strategies – would love to see more like this (i.e. for computability reductions).

How could this course be improved? Please use concrete examples where possible and provide constructive suggestions.

Comments
The course textbook could be improved significantly; the writing is unclear, and there is one chapter of the book (space complexity) that is not even written, even though we are required to know the material.
I don't have any complaints. I think everything was great!!
The latter part of the course—especially time complexity was very confusing and I feel like I could have benefitted from more examples and better instruction.

Comments
<p>It should not be mandatory for the computer science concentration. The professor genuinely had to justify the usefulness of the material, which he prefaced by saying, "Now I understand that this doesn't seem useful..."</p>
<p>Further editing of the textbook and better interconnectivity between the topics units at the end.</p>
<p>I felt like there was insufficient support when it came to going over previous psets/midterms. Often, I felt like I understood something, wrote out a full proof (that I was convinced by) only to receive 0 for it.</p>
<p>However, when I went to office hours, TFs were unable to explain to me why my solution was wrong/diagnose my conceptual misunderstanding as they hadn't marked the question. I often received vague answers of "I guess this just wasn't what they were looking for".</p>
<p>If possible, it would be very useful to provide problems that are designed to test our understanding of concepts and diagnose common misunderstandings. Although I attended many office hours and signed up for tutoring, there are still answers to psets/mid terms where I don't understand why I was wrong.</p>
<p>This final point is probably more a reflection of flaws in my approach than in the course but I felt like, no matter how much I reviewed the material and clarified my understanding or how many bonus points I scored on the pset, I still wasn't able to improve my grade on the midterm. That was very frustrating for me.</p>
<p>The problem sets should be shorter, and should align better with break periods.</p>
<p>Sections! (I'm a grad student and I taught 4+ sections in similar big science courses for the past 3 years, so I think I'm not uninformed when I complain about this.)</p>
<p>I am most shocked at how little quality control there is about these sections. The TFs are great, in the sense that they are willing to teach and knowledgeable about their subjects, but one thing about teaching is that it is an ART, not a science, and just like any kind of craft, it takes time to get better! It is obvious to me that most of the undergrad TFs has zero experience in teaching this material, and their time management is poor — I attended a total of 3 sections, at different period of the course, by different TFs —and every time they ran out of time, never managed to explain the most complex part of the course material. I was disappointed and thought the effort I made to attend was wasted.</p>
<p>This is NOT to blame the TFs, but I do blame the lack of training in HOW to teach. Also, there has to be some accountability of the TFs. Could Bok Center provide teaching seminar, and more importantly, mock teaching sessions for these TFs before they actually teach? I remember before I had to teach sections, our department mandated a whole year-long teaching training program, and during the actual teaching semester Bok Center was still providing consultance on how to teach (including video taping in class and reviewing them afterwards) — and even with that I barely thought I managed to teach anything. Now I couldn't even think of just sending undergrads to teach undergrads without any training.</p>
<p>There are other minor things that this course could improve, such as (1) actually finishing the textbook and (2) try to write out more informative, typo-free answer keys for psets, but I think the Section problem needs to be solved if the course were to improve significantly.</p>
<p>Anyway that's just my two-cents,</p>
<p>The lectures and the textbook tend to be hard to follow, often leaving me confused until I can go to office hours or section to have the concepts explained to me.</p>
<p>I found lecture to be a bit overwhelming with the emphasis on going into the details of proofs, and it ended up being a bit hard to follow and didn't really increase my understanding of content.</p>
<p>Overall I liked the way CS121 chose to introduce the material. However, I think it might have been useful to sprinkle in more content related to formal languages and automata theory, if only because this is the language used by many TCS texts to discuss the same topics.</p>
<p>There could be more effort given to making sure individual students are succeeding, as the class is so big the students seem to get overtaken by the machine that is the course.</p>
<p>This course was horribly taught. It was far too hard. Some problems were as follows: There was not nearly enough opportunity for help. Office hours run by TF's were disorganized and horribly inefficient (Just using Inline will solve much of this). Often the Problem Sets were so hard and the TF's answer sheets were so poor that they could not provide help on basic questions and concepts. Boaz is impossible to understand in lecture and extremely unorganized. A chapter was missing in the textbook that we were expected to learn somehow (Space Complexity). The Test were absurd in that it didn't test some basic knowledge but just repeated what was on Problem Sets with different mind bending problems that are not conducive to a time restrained test. There was no intermediary between the textbook and the lectures (Boaz tried to say section provided this in the midterm review but it was the students fault for not attending section, but section was barely any help as we never went through real proofs but rather just thought about ideas for some random examples). There were more problems that I had found throughout the course as well.</p>
<p>It's called "Intro to Theoretical CS," but it seems that "theoretical" just means "useless," not "theoretically interesting". Slim down the</p>

Comments
notation (don't just introduce notation because you want to, try to avoid cluttering the material), focus on interesting concepts instead of minutiae of how Turing machines are implemented and forcing students to memorize $EVAL_{\{s,p,q\}}(M,P,x)$ – that's not clear or helpful to anyone.
This class often feels hand-wavy. I didn't feel like proofs were really done right toward the first half of class, and things were too poorly defined toward the end of class for us to really prove things in the second half of class. Also, TF's don't always understand proofs that aren't identical to the solutions.
Make a separate portal for extension students on Ed or create a subgroup so that questions and posts pertaining to extension students can only be viewed by extension students. No offence intended, but the amount of "What arrangements are there for Extension Students?" or "Will these lectures/sections be recorded for Extension Students?" questions that inevitably pop up on many threads can make it more difficult for non-extension students to find relevant information. (Note: I'm not saying that they should be totally segregated – it's fine to share a platform when it comes to questions about course material, but for logistics issues, it's probably better for both extension and non-extension students if we could just keep the announcements/questions separate.)
I think section should be assigned
Complete the textbook!
Boaz is not a great lecturer; he should not do proofs on the board because they always seem like he did not plan it b/c he ends up erasing half of it as it goes and it's all around confusing. He assumed a greater level of knowledge than students actually have on theory. He should stop explaining away certain proofs as being trivial because they are not obvious to everyone. Stop assigning psets on reading week so I can actually have time to study for your exam. The way that the problem set is graded is also ridiculous because I can get more credit for certain questions by just writing "I don't know" than honestly attempting them. When you say that the advanced material is only going to be in the true or false section of the final, how about you not make it worth 70 points. The most confusing part of this class is circuits...we should spend a bit more time on that. Also, while I understand the point of the textbook and that there is a lot of material to cover, I don't think there students should just be expected to teach themselves material on the textbook that is never covered in lecture or sections, at that point there is no teaching happening. Office Hours could be a lot more helpful. That being said, there should have been a chapter in the textbook on Space Complexity because going off of solely the slides and Ed was absolutely ridiculous especially when lecture wasn't clear to begin with.
Actually having assigned sections would probably solve a lot of problems.
The lectures were not effective or interesting – and the problem sets and tests were kinda arbitrary and reflected random topics that were focused on not the material as a whole.
<ul style="list-style-type: none"> –So. Many. Things. I can't even name them all. –All of the course materials have a MILLION typos. –Homework deadlines are changed spontaneously throughout the course. –Extremely unwelcoming introduction to the CS department at Harvard that deters people of color / minorities from continuing in the field because of the stress culture it encourages
More lecturers with different styles. Boaz and Sudan's teaching styles did not work for me
Too much emphasis on understanding notation, not on understanding material; too many bonus points, not enough conceptual questions on PSets.
The textbook is incredibly dense. Perhaps eliminating the last few lectures on cryptography etc. and spacing out key material (namely distinctions and relationships between function classes) would be great. These are pretty minute, though.
The lectures were convoluted and difficult to understand at times, and office hours are not helpful because of the large queue. It is hard to understand concepts or how to structure your studying to best understand the material.
I love Ed but it's really hard to find any files on it. Having files organized into folders on Canvas would be really helpful. Also, some of the primers posted on ed — e.g. the "trick or treat guide to reductions" — were kind of the only means to understanding what was expected of us, rigor-wise, on certain problems. It would be helpful to make more of those handouts if possible (like a general outline of the steps in an amplification question) going into psets, especially since OH the week of that first randomized alg pset was pretty packed.
Some of the problem sets (I think PSET5 especially) are simply unfair in their difficulty and do not help us learn because no one can solve it without getting the answer from office hours. I appreciate when lectures give a slower, more detailed picture of specific proofs and topics rather than skimming over many different topics. I think the textbook proofs should have a similar level of rigor to that expected on the final – the "Proof Idea" is too general and the full proof too technical.
It would be great if the course staff published some useful resources (such as the reduction guide) on the website to emphasize their importance. More of the material in the textbook should be covered in lecture. Most importantly, more CAs should be hired since there were often not enough of them at section to help everyone.
The course is entirely reliant on office hours, which is fine, many CS courses at Harvard set difficult and long problem sets which often require hints from teaching staff, but on the harder problem sets I would have to go to about 4 sets of office hours during the week (each 2 hours long), with more than 30 students between 2 TFs. There's no way the TFs could answer all the individual questions of all students, as such I maybe got a total of 10 minutes of question time over the 8 hour s.

Comments
Instead, I would suggest that the office hours are structured during the week so that say, monday night focuses on questions 1 & 2, and tuesday 3&4. By doing this TFs can focus on perfecting the explanation of fewer questions and students (who will most likely want to hear similar explanations) can get their questions answered simultaneously.
Some of the grading can be a little inconsistent, but they do their level–best and there is a regrade option that is pretty easy to use.
We need MORE support. The textbook is cumbersome to read, and TFs are often inaccessible since office hours are so crowded. There has to be more done that is beyond just Boaz's lectures, which also need to be done at a slower pace.
More consistent grading schemes
There should be a chapter in the book on space complexity. Also, it would be nice if the collaboration policy were a little more lenient; as far as I could tell, the collaboration policy doesn't allow you to check the answers to PSets with other students.
lectures are usually very confusing – improve terminology; does not cater towards students who do not have extensive math background – explain concepts in English terms after explaining them in math terms; textbook can be super confusing – get new textbook
Incorporate sample problems similar to those in the problem sets into the lecture.
Boaz is a smart guy who cares a lot about his students, but his lectures can be really tough to follow, same with the textbook. Psets can be really hard to even understand
At times, the lectures were a bit hard to follow because of slightly longer proofs (in particular I am reminded of the proof of godels incompleteness theorem). While this is somewhat due to the nature of the course material, it felt like there was some room for improvement.
The textbook that the course is based off of is very much a work in progress, is organized sloppily, and is very difficult to navigate. It omits a lot of important information (space computation) and proofs (the randomization chapters come to mind). The lectures also aren't great and can often confuse me even if I've done the textbook readings beforehand. Also, the psets are ridiculous— I find that they often test brute algebraic manipulation or skills that are completely unapplicable to the class, or that they ask the same type of question ad nauseam so that they just feel like busy work. The incompetency of some members of the course staff don't help— office hours often end up as a waste of time, usually with course staff re–explaining to me what I already know and not addressing my actual question. I would much appreciate it if some members on the course staff were able to just say they didn't know how to explain a pset problem, since it saves everyone's time that way. I find myself just directly messaging course staff who I know are reliable, which seems to defeat the point of section or office hours entirely. Also, grading leaves much to be desired— I feel like it's often rushed and while some CAs are very receptive to explaining why I got things wrong, many more seem to get defensive about regrade requests or even questions posed in office hours. I also get the sense that CAs frequently neglect to read my whole proof, or even immediately mark proofs as wrong when it doesn't adhere strictly to the answer key, even though there can often be alternate correct solutions. Again, this is not to say that all CAs and TFs are like this. Many are really great and helpful, but the fact that there is such inconsistency in the group of CAs is a problem. I find myself gravitating towards the CAs with more mathematical maturity, and to be frank, I think CAs in the future for this class need to be held to this standard or else they actually detract from my learning in this class.
Too many problem sets in too close proximity towards the end of the semester (especially that one that was due the Sunday of break)
The content of this class is honestly quite difficult and unintuitive, so more resources (in addition to the textbook), such as short clips and guides (like the reduction guide) would be much appreciated.
The second midterm was incredibly unnecessarily stressful, I'm begging you to make it less stressful or at least warn students ahead of time that they probably won't finish.
Consistency in the course curriculum could have been better, as the definitions and theorems relating to certain topics were debated between staff at times. Additionally, more prior knowledge regarding assignments (i.e. whether there would be more than one programming assignment, the grading rubric for the project, etc.) would have been helpful.
the textbook could be better
Total time spent on problem sets is too much if you want to actually understand the solutions.
Lectures are too big / not that interesting
Better lectures.
– Boaz clearly knows a lot but is not an effective lecturer. Lectures moved way too fast and concepts were not explained thoroughly – I was often lost after 10 minutes and couldn't recover. I learned most things from the textbook, which I had to read multiple times over to extract information from. – TFs often know the subject matter but are rarely good at conveying information in a way that is not scary and stress–inducing – problem sets are so unreasonably difficult that I have no hope of answering a single question by myself, even if I were to think about it for hours (which I have tried). They were way harder than exams, and the subject matter often bore very little resemblance to what we learned in lecture or in the textbook. – this class made me feel beyond incompetent and decreased what little interest I had in theoretical computer science to zero. I am

Comments
very unencouraged to continue in the computer science concentration but I have resolved myself to having a painful experience. – there needs to be more final prep, especially if Boaz happens to lose the practice solutions again
The homework is sometimes really bizarre. The grading is inconsistent amongst TFs. The errors in the book do not bother me, though it would have been helpful to have a section on space complexity since it was covered. The sections really vary in quality between TFs, though otherwise it was pretty good.
The textbook is horrendous. Not only is it incomplete and prone to errors, but incredibly difficult to parse. We were provided an alternate reading on poly-time reductions from the Berkeley textbook, and it was the first time that readings were useful in this course. The only redeeming portion of the textbook is the occasional "Proof Summary" sections. The lectures in this class face the common issue where the professor is far too smart and is unable to relate to the academic level of his students. Therefore, the lectures often move way too quickly and assume knowledge that the students do not have. The structure of the problem sets is odd because many of the problems require intimate knowledge of the material that students would be unable to figure out without consulting the TFs. This is not inherently an issue, but when the OH are overwhelmed with students for far too many TFs then it becomes a problem. Similarly, I find it odd that the problem sets contain bonus problems that often challenging or require advanced knowledge (such as probability or high-level mathematics that is not a pre-rec for the course). When this is the case, then the bonus problems only serve to disadvantage students that are already struggling and create a larger gap between those students who are comfortable with the material and those who are struggling.
Better organization, better textbook, more support for students who aren't as strong.
I would've liked to see more proofs done in class that related to the homework problems.
There felt like there was a large gap between what we learned in class/read and applying it in the homework.
material can get rather dry
All the assignments at the end (the last two psets and the final) felt very cramped and stressful, especially given Thanksgiving break. Would have loved one fewer pset (or if the last pset was all bonus points). Also I disliked Ed, much prefer Piazza.
The lectures contained a lot of information that we never needed to know, and all of these extra details seemed to cloud the main points. In addition, it was a little frustrating that sometimes the psets were undoable without a hint on edstem (and in some cases going to office hours would be the only way one could even know how to approach the question).
The textbook is still a WIP, and can be difficult to understand at times. Lecture is sometimes confusing. Problem sets range from easy to very difficult, with a lot of variance.
There are frequent typos in the textbook and problem sets, which are confusing especially when the TFs don't know the material really well either.
Please have a finished textbook (one of the chapters was completely blank and we had to rely on section notes that had some errors and a single lecture). Slow down lecture pace – lecture often went too quick to absorb the material. Give TFs a more thorough answer key to use for psets during office hours because there were too many times where they didn't understand the pset question or couldn't help. On exams don't have built in extra credit, just curve. Having built in extra credit just skews the scores and most of the benefit goes to students who are already doing well in the class.
More office hours during the day rather than at night
More support for struggling students. I did very poorly on the first midterm and wasn't really sure how to re-approach after that. I think I got less engaged and fell a bit through the cracks of the support system.
The textbook is lengthy and has extensive math which isn't relevant to the tested course material.
Textbook examples and wording could be made more clear. (For example, hand drawn examples in the textbook should be remade into well-formed diagrams to be made clearer.)
The spacing of problem sets at the end could have been done better
I thought the course was generally well structured but I didn't find it very interesting overall. I would have appreciated going a little faster through the middle of the course (for instance, not spending a whole lecture proving 3SAT is NP-complete) so we could spend more time on the interesting more advanced topics that we touched on at the end. Additionally, the finite computation material was a bit dry, so I might have liked a little less time spent on that.
course is pretty disorganized in general (1) homework is extremely frustrating and tedious – the pages of induction proofs on hw 1, etc. – there definitely is a way to ask hard questions without making it so painful to do. For the probability homework, it was so painful having to 12 mini 5 point questions on silly things like find a X and Y that are dependent st $E(XY) = (EX)(EY)$. Look at stat 110 homework as a guide! They have problems that don't require that much grinding / bashing (in fact, the opposite) but beautifully illustrate the concepts. If the homework is so hard that TFs cannot do it without looking at the answers, that is telling you that something is wrong. Also it's a problem if a lot of the TFs need to ask questions on slack for clarification. (2) TFs have to do so much work for the class (and they are busy people)! This makes all the midterm/finals answers (and important homework answers) delayed till like 2 days before the midterm.

Comments
(3) midterm/final prep – see above for midterm/final prep being so disorganized. Also it's unclear how to study for midterm/final. For stat 110, there's a clear study plan – you do the 8–10 practice tests released and you're set. For cs 121 – you do the 1–2 practice tests, and the midterm prep, write the cheat sheet and you still don't feel prepared and are not prepared when you do the midterm. I just feel constant despair for this class when I prepare for midterm/finals.
(4) lectures and textbook are really confusing and extremely dense. Feels like a rush to write things down and go through the slides. The expectation to read the entire textbook is ridiculous – I did that for the first 2 weeks and it took me 3 hours to read a chapter. I couldn't maintain this as my other classes picked up and textbook got harder and harder.
Course materials are wrought with typos and course staff seems to emphasize a lot of nuance and detail required in the course all while being unable to deal with it themselves.
Lectures are hard to follow, psets were extremely difficult and time consuming, office hours were always extremely crowded
Near the end of the semester assignments felt very rushed. I think it's very unfair to have a pset due sunday of thanksgiving break. Also to have an assignment due during reading period after this is really asking too much.
Textbook was badly written and hard to parse to find actually important information. I think having something more informative than the section notes but less dense than the textbook could be really helpful to the course. The problem sets were often incredibly long (especially the first 2 were both long and hard to do because they were required higher knowledge of random topics and problems not taught in class)
Lectures could be very confusing especially when digging into detailed proofs.
Pacing of lectures was a little bit slow, sometimes. As a result, the professor had to rush through the later slides which made it almost impossible to follow.
Don't make the psets take more time than my job.
Some of the assignments seemed pointless or unnecessarily tedious. Midterm 2 was a ridiculous time crunch. There was a LOT of material and some of it seemed unnecessary, Boaz could have done a better job of emphasizing what was most important for us to understand.
I felt that the course emphasized knowing a lot of facts at a very superficial level instead of understanding their implications at a deeper level.
In a lot of ways. First of all, I don't think the textbook or lectures are very effective. The textbook goes into an extreme amount of mathematical detail that is not worth working through because assignments and exams expect nowhere near that level of rigor. Additionally, the lectures weren't very engaging.
I'm not really sure, seems to be that this class is great for people who get it and terrible for those who don't.
Finishing the chapters of the textbook on space complexity and proof systems.
– The textbook could be more developed (there are some missing or in-progress chapters). – Need a little more standardization between TFs' grading styles and levels of harshness.
The materials provided in the course can be improved as they were often unreliable, making learning extremely difficult and tedious. For example, the midterm prep solutions. Over four different versions were released. Or for example the Midterm II solutions which “would not necessarily receive full credit.” I get that having unreliable course materials can be arguably good since it is sort of like an exercise in “testing your understanding” to constantly be questioning the material, but when the material is already challenging it's frustrating and inconducive to learning.
The proofs in lecture can also be improved. There were several times when a proof was presented too quickly for an undergraduate introduction to theoretical computer science course or were even admittedly “botched” in their presentation. Perhaps writing down the proof beforehand and ensuring that it is accurate would be helpful?
It would be great to have more consistent levels of difficulty among psets, midterms, and finals. Personally, I found that midterm II was unreasonably hard whereas the final was significantly easier (e.g., the reduction in the final was much much easier than any of the reductions in Midterm II). I'm struggling to find a pedagogical reason why this inconsistency in difficulty might be good. In short, perhaps standardizing the level of difficulty.
Finally, I think that background in formal math should be required (CS20) as a pre-req. I didn't take CS20 and did not self-study over the summer (which was dumb on my part), and as a result struggled a lot throughout the course.
–assignments/prep materials/lecture materials/textbook need to be SIGNIFICANTLY cleaned up/improved before releasing to students. It is really unfair and impedes students learning when they are released with significant mistakes (i.e. discord between textbook and lectures, mistakes in the exam review guide that are corrected 2 hours before the exam after being found by students, etc.) –prevent massive schedule changes/disorganization. For a course of this size, there is no reason the pset schedule should be significantly changing around at the last minute (such as when the post-thanksgiving pset was changed to be due a week and a half earlier).

Comments
<ul style="list-style-type: none"> –consolidate course materials into one easily findable location. It's really inconvenient to have to check Ed, canvas, a course website, and gradescope to figure out what things are due and where to submit them –grading is really inconsistent and should be standardized across TFs
More Quantum! Not much else, it was thoroughly enjoyable.
The lectures were sometimes not too helpful because they moved at a somewhat slow pace and didn't cover all the material in the book.
By getting rid of the weekly quiz! I strongly believe students are hammered enough with weekly assignments, adding anything additional to that is rough.
So much work got pushed to the end of the semester. Everything after the second midterm felt so condensed. Up until the final, it felt like we were constantly psetting. The work was helpful, so it was good to do, but spacing things out more would have significantly decreased the overall stress and sleeplessness of those few weeks.
<ul style="list-style-type: none"> –More consistency in difficulty of problem sets and exams. It is really frustrating to not be able to predict how much time an assignment will take. –It would have been helpful to have a textbook chapter on space complexity –It would be nice to have more opportunities to learn about problem-solving strategies. The lectures and textbook were very theoretical, so it was tough especially at the beginning of the semester when looking at a problem to think of ways to attempt it. –Office hours were usually not helpful.
I didn't enjoy it, but not sure what can be changed. It's core material. :/
Too much irrelevant content (about philosophy and stuff) in the textbook — plus the textbook changes mid-course. It is especially annoying when you are doing problem sets and theorem numbers change in the textbook (I printed half the book out and was outraged when the page numbers don't fit).
Could be more explicit about what materials in the textbook are on the exams too.
Space complexity was very unclear, and the entire P–NP problem came about without a lot of lecture-based explanation. The final quarter of the class, on cryptography and quantum computing, was difficult. The worst thing, though, was the homework. Homework assignments were too long, as they often involved waiting for material to come up in lecture and left the Q&A part of section feeling very difficult. The homework had you constantly trying to work on 121 without enough information, and then, for the last 3 days before due date, rushing to complete it having learned how to at the last minute.
More structure lectures.
Material was often confusing and not engaging to me personally, had to rely a lot on the textbook, which I didn't find to be a super helpful/clear resource
The material itself is quite dry, but I really do think the course made it about as palatable as theoretical computation can be.
The textbook for CS121 is difficult to learn from and could benefit from an increase in use of examples. Often, the side-bar notes next to graphs that were written in lamens terms were the most educational for me as a student.
The textbook still needs a significant amount of work before it can be relied upon.
The lectures sometimes feel a little underprepared. For example, some of the proofs given in class are a little hard to follow.
Problem sets shortened or simplified.
The problem sets got very monotonous by the end: reductions take a ton of time, and while they are a valuable skill to learn, the writing required is not proportional to the thought required; it often takes no more than 5 minutes to figure out how 3SAT <p F, but it can take half an hour to hit all of the required points for a reduction question.
Some exam questions worth many points required a specific piece of knowledge that if you missed for some reason you would not be able to work out the solution
Divide the course into two semesters and spend twice as much time on each topic to create true understanding of the material and effective learning without expecting too much out of a student. Otherwise, the course's material seems too thick and difficult to manage with active participation in other classes, activities and health.
I would like to see more worked-out examples similar to the homework problems. For example, the "Trick or Treat" NP–reductions guide that was released halfway through the semester was very useful for my understanding of NP–reductions. More handouts like that would've been really useful.
Standardization across sections.
Honestly it overlapped a lot with CS124 so it was pretty boring to take after 124. It would be good if these classes covered different material. Exams were fair.
I'll start with the textbook. Boaz insists on teaching this course from his textbook, which is far from a finished product. Rampant formatting issues, hand-drawn diagrams, entire sections that only say TODO, and a whole host of other issues make this textbook incredibly hard to parse. Furthermore, for whatever reason, Boaz chooses to explain concepts in the most complicated way possible, and assumes very different levels of background for different topics. This tendency to overcomplicate concepts continues

Comments
through lecture, which gives students the impression that they are in way over their heads. Basically, I only ever learned a concept when I used non-course materials. Boaz's instruction is woefully inadequate, and I think this course would benefit greatly if it were taught even by other course staff (Will, Madhu).
I think the beginning of the course was too slow and detracted from time that we could use to make more sense of some of the harder material. I didn't like the idea of mandatory section either
The teaching in this course needs work. The textbook is riddled with incorrect statements and unclear statements. The course would be improved 100x if it just used Sipser's book until Boaz finishes with his book, because it's almost unusable in its current state and really cripples the course.
Teaching from an established textbook and generally following more established methods for an introduction the theoretical cs class would make the material easier to understand.
I can see how the problem sets relate to material learned in class and from readings, but I feel lost as to how to answer them. Maybe in section we could spend more time going over practice problems as a class
I did not really feel supported all the time. Going to office hours was often unproductive, except for one specific office hours that I would always go to. Material can be very confusing and intimidating, so I think finding more ways to get students to interact with the material would be extremely helpful. I found the psets to be pretty hard, and took longer than expected. One or two extra late days would be helpful.
Pset 5 was much harder than any other and I think it is good to have all pset on similar level, so people plan accordingly with their time.
It would be helpful to spend a lecture going over various proof strategies (induction, contradiction, etc.) and explaining what is required in a CS 121 proof vs a more formal mathematical proof. It would also be helpful to present problem-solving strategies more explicitly (i.e. to prove a function is computable, we should describe the algorithm that computes it). One difficulty I had with exam questions was that you had to first determine if something was computable or not (or in NP or not in NP), then prove it. If you got the initial judgement wrong, then you got 0 points for the entire question. However, PSETs would tell you whether or not a function was computable (or in NP), then just ask you to prove it. If exams are going to ask you to make a judgement, then PSETs should provide practice doing so as well.

Requirements Comments - What did this course require of you?

In your opinion, what preparation or background is necessary to take this course?

Comments
Some background with mathematical proofs
Writing proofs.
Some discrete math background
I think anyone can take this course with some background in basic proofs. No rigorous math/proof background is required.
CS20, CS50, CS51
Discrete mathematics.
Mathematical maturity (some coding helps but is not needed).
Comfort reading mathematically dense texts will help but it isn't necessary.
CS50, or some coding experience. Math proofs.
The course website suggested following an MIT course — that's the only prep I did, and it seems sufficient. (Never took discrete math or proof-based math before)
CS20, or some background in writing mathematical proofs, as well as basic knowledge of CS, like CS50.
CS20 was helpful towards the beginning of the course, but definitely isn't necessary.
I don't think any preparation is particularly necessary. Some familiarity with writing proofs is helpful, but the course gives you plenty of resources to learn this.
CS20
CS20 or some sort of proofs class. Probability class might help as well.
Honestly nothing, any proof-based math class makes this class trivial (math 25, CS 124, etc).
Some exposure to computer science.
We spend enough time on intro to discrete math and proofs that it seems like you shouldn't need much background, but the first few weeks had a lot of problems (Boaz's proof of Cantor's diagonalization was incomplete, I remember thinking the first few

Comments
lectures were very bad but I don't remember enough to give specific examples other than the diagonalization proof) and it seemed like people without a discreet math background were struggling.
Having a strong mathematics background helps greatly (especially knowledge of proofs). Knowing CS is, in fact, less essential.
CS20
Problem solving background
this is an intro class....but come in knowing how to write proofs
Proof-based math. Some discrete math/probability probably helpful.
CS20 or other formal proof writing but can also learn thru this course.
You're going to be teaching yourself the whole course from the very beginning so start whenever even if that includes learning the material beforehand.
Prior theoretical CS and even some prior knowledge of graph theory
Some CS and proof-based math
Discreet math (at least familiarity with sets) and some sort of computer science (for familiarity with time complexity).
Linear algebra or statistics would be recommended
A math class that familiarizes you with basic proof techniques (CS20 and/or Math 23+)
Some mathematical maturity
You need a background in proof and discrete math. CS20 is super helpful.
If you're very smart, not that much – there's minimal coding background required and proofs can be learned on the spot. But for most students, CS50 and 51, as well as some proof-based mathematics.
Statistics at STAT110 and proof theory at CS20, perhaps some background reading as well.
I walked in without a proof background and it turned out fine! You just need to put in a little work at the beginning.
Not even sure, but a lot
A strong proof based math background helps alot.
Discrete math, proof-based math classes
some proofs
Mathematical proofs
Background with math proofs
CS 20
Good proofs intuition, discrete math/combinatorics helps of course. Surprisingly you should be good at algebra to follow a lot of the proofs in class and manipulate bounds.
A familiarity with proofs and discrete mathematics is useful, though I think you could pick up the necessary skills as you go along if you had to.
Some level of familiarity with basic set theory, mathematical notation, and proofs will help significantly. Otherwise you don't need anything going in, not even CS50.
Proof background
A strong background in mathematics and the willingness to work hard.
A strong math background and preferably some coding experience.
Some proof-based knowledge would be helpful.
proof background
No preparation necessary, but one needs a willingness to accept and learn the foreign nature of mathematical language and computer science thought.
Proof based math, minimal but nonzero coding experience
CS 20.
CS20 was very helpful.
Significant prior proof experience. Interest in theoretical computer science. Programming experience so you understand why we're studying the things we are.
As much math as possible. One should not be afraid of having to decipher tons of math symbols and parse through heavy proofs.
CS20, Probability (Stat 110) would be helpful
Proof based math.

Comments
CS20
It is helpful to have experience with proofs but not necessary.
CS20/similar proof and logic experience
proof based math background in particular discrete math would be helpful
I didn't take CS20 but learned it over the summer, and I did fine.
A LOT OF MATH. at least take cs20
Experience with discrete math– CS 20 is nice, but self study for a few weeks should be more than enough.
Math proof experience and basic programming is enough
Knowledge of proofs and good athematical sense
CS20, Discrete Math
Some mathematical maturity is necessary. It would be very helpful to be confident with probability theory before taking this class. It is absolutely necessary to be willing to think in new and different ways. This class was significantly easier after taking CS 124. It is easier to understand the course material if you are well–rested.
Stat110
I took CS20. Although I found my CS20 background indispensable, I really wish I'd had even more of a background in proof–based math.
CS 20 level
Familiarity with induction and proofs. Knowledge of surjective / injective functions. Math 23 was good mathematical preparation.
Probability is helpful. cs50 or any intro course
Some mathematical proof background, at least a little exposure to programming (if not to help with comprehension, then to make the material seem a little more interesting and relevant).
CS20
CS20 material – mathematical maturity like induction, probability, graphs etc.
Some proof experience, or at least some basic math.
CS20 definitely
A strong math proof background would be helpful.
Some discrete math and proof–writing strategies
CS20 or experience writing proofs
Background with proofs is essential to succeeding in this course! Also, it would be helpful to have some prior knowledge of probability and statistics, but certainly not necessary.
Proof–based math experience
Comfort with abstract mathematical formulae, some familiarity with basic programming principles
Mathematical and logical intuition is the most important thing – programming knowledge is helpful but not a must. Proof experience would be helpful too.
None really – just some basic mathematical/CS intuition
Be smart.
Some experience in proof based math makes this so much easier.
Experience with proofs and basic number theory.
CS20 or formal math background.
CS20 (or familiarity with proofs), Stat110, CS124 is helpful
Basic proof skills, basic code experience
Mathematical background in proof–writing
Basic understanding of proofs – can learn this on the fly
A solid math background. Knowledge/experience with proofs and basic CS theory (like boolean algebra and big–O) is very helpful.
–A tiny bit of programming and probability experience
–Math at the level of CS20 at a minimum, although I did well in CS20 and still found this course extremely difficult.
Discrete math and some probability
Proof based math class

Comments
Comfortable with proof (MATH 25A is sufficient), some knowledge about algorithms and data structures would be beneficial too.
Probably CS20—even other proof-based math is not necessarily great prep.
Discrete math, set theory
No background
Some familiarity with proofs
CS50, CS51, and CS124 would really help, if you actually take them out of order.
I came in without having taken cs20 or a proof-based math class and ended up fine! As long as the student has some level of mathematical maturity and possibly a brief background in probability might help.
Proof-based math and probability.
A mathematical background helps.
Previous exposure to proof-based math.
Any sort of proof based math course
Mathematical Proofs, Real Analysis, Probability
Strong mathematical background.
Comfort with proofs (most important), math/CS (important), and probability (least important but still useful).
CS50 / introductory proof experience.
some math, really not a lot.
Some experience with proofs, but I think anyone could take it if they make sure to put in effort early on to make sure that they learn how to do proofs well
CS 20 is very helpful but not necessary. I would recommend having a basic understanding of writing proofs, proof techniques, and some general familiarity with big-O, probability, and sets.
Mathematical proofs
A slight math background, like Math 21 would be helpful. Stat 110 is essential for one chapter of course and otherwise unrelated.
Experience with proofs.
cs20, stat110 (not necessary but very helpful)
CS20, plus you need to be familiar with a lot of the concepts already if you want to do well. If you go in expecting it to be a true introduction, it will be a very difficult course because of how fast it moves.
Being comfortable with proofs.

Recommendations Comments - Would you recommend this course?

What would you like to tell future students about this class? (Your response to this question may be published anonymously.)

Comments
As a person of color, I cannot even begin to describe the extent to which I felt belittled and unwelcome in this course (and I feel like the staff will not even allow others to see this post to cover up their mistakes—you cannot silence us and the truth forever!). Computer Science is a field that is notoriously unwelcoming to minorities, and I can say that this course did nothing but affirm this so-called stereotype. Not only do the staff sometimes not even know what they are saying, but they are strangely both arrogant and insecure and defensive at the same time. They talk down to me as if I do not understand anything, but when I ask a question, they seem to take it as an attack on themselves and respond very rudely. I feel like the material of this course is interesting, and this course is actually my highest grade right now, but the teaching staff single-handedly made this course the worst class I have ever taken, made me feel beyond unwelcome, and confirmed that Computer Science is not the field for me.
Amazing amazing class! Best one I have taken so far. The book is super helpful, the weekly quizzes and psets are very reasonable and make sure that you don't get lost, the CS 121.5 sections were amazing, and the material is REALLY cool. I loved everything about this class.
It was quite an interesting class. Very well organized with a very hard working teaching staff. Some Psets are difficult, but the bonus points more than made up for it.
For CS, it's a requirement so there's really no choice. It's great stuff, you get really a solid background in computation which sets you up well for an algorithms class like 124. For-CS, take it if you are really interested in learning about the limits of humans computational powers and the things we have and have not been able to achieve in that regard.

Comments
<p>This class is a nightmare. The textbook, while thorough, is exceedingly verbose and convoluted. Lectures are difficult to parse. This is a personal opinion, but the material is simply uninteresting and inapplicable. The one thing I can say for this course is that Boaz cares about the students.</p>
<p>It's very much a theory course. If your goal in life is to do SWE you will probably have a bad time. However, if you enjoy theory or math you will find the content to be very interesting and worthwhile. Also don't listen to the bad things about NAND, it's elegant once you get used to it.</p>
<p>I wouldn't believe all the negative hype about this class. I heard terrible things about this class but it wasn't as bad as the rumours made it out to be.</p>
<p>The toughest part of this class is understanding all the concepts. Especially since the textbook doesn't explain them very well and has a lot of details/digressions that don't end up ever mattering on psets or midterms. I think Boaz hints at that with his emphasis on focusing on big concepts but I just wanted to explicitly say it.</p>
<p>Also, I found it almost impossible to understand why I was wrong in this class. You think you understand the concepts, write a proof that makes sense, and then get a 0. Your ability to do the psets also isn't very indicative of your midterm performance. That was the most frustrating part for me.</p>
<p>This course is difficult, but it is well-taught and you have support. Find people to work with, read the textbook, and go to section, and you'll be fine.</p>
<p>What I really really like about this course is that it is the only course I've taken at Harvard that had given me a glimpse of the "edge" of human knowledge. After the first midterm, we constantly encounter conjectures that we have no idea how to prove — and man that gives me an intellectual orgasm! Usually in a course you learn what people have figured out 100+ years ago, there's a bit of that in the course, but you also learn what people have yet to figure out for the 100+ years ahead, and that is exciting!</p>
<p>Also, the online support forum is superb. Boaz and all the TF actually care about us learning — really, learning about critical thinking, asking good questions and understanding complex propositions — not in the sense of solving the pset at hand. That is something to applaud to!</p>
<p>This course takes a lot of effort, so make use of office hours and section, and always ask questions when you don't understand something, as it is very easy to just glaze over an important topic.</p>
<p>Although my main reason for taking this class was initially that it's a CS concentration requirement and good prep for 124, looking back on the semester, I came away thinking about computation differently. Lecture can get into details and be a bit overwhelming, but if you make sure you understand the high-level takeaways, you'll be fine. There's a lot of support to make sure you get psets done and understand concepts as well! Definitely a lot of work, but was rewarding at the end.</p>
<p>If you're a CS concentrator you have to take this class, but that's not a bad thing! This course is a solid introduction to theoretical computer science. While it is still a very serious course, it does a good job preparing students with less background for future theory courses. Although others may disagree, I personally think Boaz is a very good teacher and clearly cares a lot about the class.</p>
<p>The workload is quite reasonable, and I think it has been significantly reduced from previous years. The problem sets are generally pretty interesting, and have bonus questions that are more difficult and fun to think about. Towards the end of the class writing up solutions can become a long and tedious process, but such is life in any course expecting rigorous proofs.</p>
<p>If you've heard something that makes you scared of this course, don't be. The teaching staff really doesn't want to make this course stressful, and provides lots of accommodations (bonus points on tests and psets, bonus projects, lots of study resources) to reduce stress. They ultimately just want you to develop an appreciation for CS theory and some experience writing formal proofs, and you will definitely gain these things if you put effort into this class.</p>
<p>While professors like Jelani Nelson and Michael Mitzenmacher are insensitive and more or less own the fact that they are, Boaz differs from the two in that he is completely unaware of the effect his course is having on students. When he first started teaching this course in 2017, he had a "free pass," since it was his first year and the kinks needed to be worked out. Then, in his second year teaching the course, things did not really get much better. We are now in the third iteration of CS 121. There is NO excuse for the typos in the textbook. Boaz thinks it's funny that we catch his typos. Watching him stumble through teaching elementary CS material was embarrassing. He would frequently tell us that such and such material wasn't on an exam, and then proceed to put that material as the majority. Problem sets were overly long and difficult (to the point where a TF actually took a picture of how big his office hours were because they were so swamped). We would frequently correct Boaz in lecture because he's so awkward and clumsy in how he teaches that he failed to explain even the most basic of concepts. And then he "lost" the solutions for last year's final, so we basically had no material with which to practice. This course ruined computer science for me.</p>
<p>This class is very picky with their proof structure, and looks to take points off in the grading rubric rather than award them. It is easy to get lost in the machine of the course, so make sure you go to section and find a group of people to pset with early. Speaking of psets, start them early!! Latexing them is tricky given the vast array of symbols, and you don't want that to be why you lose points!</p>
<p>Horribly taught class. The content is important but the class overall is terrible. Would not recommend unless for a concentration or you are extremely interested in theoretical computation (and you don't mind proofs).</p>

Comments
<p>Honestly, you're going to take this class because it's required. It's one of the most frustrating classes I've taken here. The class itself isn't very hard, but if you have a CS background and have taken 124, being forced to take this class is a huge waste of your time. I didn't learn any interesting concepts or ways of looking at things, all I learned was how to do the minimum amount of textbook reading to decipher some terrible, overcomplicated notation and how to memorize definitions as that's the only challenge on the exams. The teaching staff are great and all genuinely try to help, which makes me wonder how there's so much fluff that they teach. Boaz was one of the highlights, and is a wonderful person, but I've heard he's going on sabbatical next year so you won't even have that. I wouldn't recommend this class to anyone, ever, but since it's required and you're gonna take it no matter what the Q guide says, good luck.</p>
<p>You learn a few topics, and a bit of it is interesting. There's a lot constructed (NAND-TM and NAND-RAM machines) without a huge amount of payoff.</p>
<p>I can probably only speak on behalf of the more math-inclined students, but if you enjoy Mathematics (at least as much as you enjoy CS), then CS121 will be a relatively easy and interesting class for you. Prof Boaz is really dedicated to improving the course experience and constantly works towards making sure that all students receive the help that they need. The textbook is also a great resource. Lectures aren't particularly necessary (especially with the textbook and recordings), but Boaz tries to make the lectures fun too. Overall, as long as you diligently complete the psets (which aren't too bad actually – there's only 1 pset every 2 weeks and you have many late days so time pressure is definitely not an issue), the course should be rather easy-going.</p>
<p>Great course in terms of interesting topics, decently well taught with a lot of care and support in terms of pset leniency and mental health awareness, especially in a cs course. Would recommend despite the bad rep</p>
<p>While Boaz is nice as a person and everything (I'm sure he is a great advisor too) and he really wants people to pass his class, he is a bad, like terrible, teacher. Yes, the material is difficult and all that, but good pedagogy would alleviate some of the struggle from that. Not only that but the Patel Fellow was this year was useless, first thing I did for this class was reach out for support and she did absolutely none of that, and office hours are unhelpful. There is a lot of material in the textbook that is not ever covered in lecture or sections and you are expected to magically understand the textbook. The textbook is terrible, goes on long winding technically difficult explanations that make readers more confused. Google has been a better resource for this class than the class itself.</p>
<p>People have a hard time with this class because they don't have a strong enough math background. I want to emphasize: I'm a math concentrator, and this IS a math class. You're going to have a hard time if you can't learn to think mathematically. Not having required sections is kind of weird for a theoretical class like this, go to them if you can.</p>
<p>The course and material is interesting but be prepared to spend a lot of time on it. If you work hard the course is manageable.</p>
<p>There are SO many things wrong with this class that the only reason people still take it is because it's a requirement. And that's probably why it's a requirement. Let me tell you more about my qualms with the course:</p> <ul style="list-style-type: none"> –All of the course materials have a MILLION typos. And I'm not talking minor misspellings / occasional off-by-one errors. These are typos that actually affect students' learning / understanding of the course material. I'd so often read through the textbook / lecture notes / section notes / solutions to literally ANYTHING from the course, find an equation or piece of logic that conflicted with my understanding of the material, and figure out after 30 minutes of frustration and confusion that it was an error. Sure, you could say (and Boaz has literally said this to students) that finding errors helps with my ~learning~ and ~depth of understanding~, but if you're like me and learning these things for the first time, all this does is create endless frustration and a shaky foundation for the rest of the CS courses you'll take in the future. Do NOT take this class. It's a waste of time. –Homework deadlines are changed spontaneously throughout the course. Policies that make you question whether Boaz really cares about the mental health of his students are constantly being enacted. –I did well in this course. I had friends who did well in this course. And yet we hated it. All I'm gonna say is – I think it really tells you something about the quality of a course if you're doing well in it and you still hate it. –And finally, this course is an extremely unwelcoming introduction to the CS department at Harvard that deters people of color / minorities from continuing in the field. Everything is taught TERRIBLY, and you will likely not want to continue in CS after the course if you have had little experience in the field before. –tl;dr, please take Sipser's course at MIT. Taking this course at Harvard was the biggest regret of my semester.
<p>Very tough, material is dense, and lectures are not easy to understand.</p>
<p>This class is not worthwhile, as it for the most part entails wading through notation and trying to figure out what is actually being asked. The textbook is also extremely dense, and very unhelpfully unclear on many important topics. Most TFs were unable to help with PSets due to lack of training and absurd difficulty of some questions, but this didn't matter because of bonus points which ensures that basically everyone gets 100 on them. The main issue in the course has to be an absurd notion of "rigor"; some proofs have to be extremely rigorous, while others are remarkably "hand-wavy" with no direction about which to do. Lectures are generally unhelpful, as difficult concepts and proofs are not explained in intuitive or helpful ways. TFs range from doing the homework with you to having no understanding of how to approach problems, as was the case for several PSets. The only redeeming factor is that Boaz is a great guy, and his commitment to openness is very respectable; he is not, however, the one to be teaching this course.</p>
<p>This class is very hard, but the exams are very fair and the support system is great. This should be your #1 priority class over the course of the semester, and even then it will still be pretty difficult. However, feel free to reach out to Boaz or head TFs if you are</p>

Comments
falling behind, and they truly try their best to help. Read the book and make sure you understand how to do everything on the PSETs. This is not a class you can skate by with trivial understanding in.
I honestly would recommend taking the version at MIT. This course is a requirement, and the material is difficult. These facts are uncontrollable. What is controllable is what course you decide to take or how to structure your time if you choose to take CS121. This course is still relatively new, so there are a lot of kinks to work out. It is very difficult to understand or learn concepts through lecture because the way things are taught are quite convoluted. Sometimes, it seems like even the professor is confused about the material. The psets are also nearly impossible without office hours, but office hours are unstructured and crowded, so it can be hard to find help. They take hours and hours because the material is so abstract and difficult to understand. I honestly don't know what it takes to understand it enough to do well in the course. Seek help early and often. Figure out what you don't know and hone in on your weaknesses. What is one of the only good things about this course is the kindness, warmth, and generosity of Boaz and the course staff and their willingness to invest time and effort into your success. Boaz's lectures might not be the best, but his capacity to help and offer support is immense. He wants to see you succeed. Utilize this, and reach out early.
I really loved this class! I was pretty neutral about it the first few weeks, but the material definitely gets a lot more interesting from there. This isn't a class you can BS — if you want to do well but are new to the material, be prepared to invest time in reading the book and going to section/OH. Psets are doable but not at the last minute and for many not on their own; if you collaborate, go to OH, or work through them carefully, they're fine and are the best way to get comfortable for the material. I wouldn't worry too much about grading — psets, the midterms, and the final have a ton of bonus points and you get several pset late days as well. If you mess up a pset or two or an exam, you can still make up for it, and Boaz is always making tweaks to the class to reduce student stress and help you succeed.
Lovely class. Boaz is a genius.
CS121 is definitely a difficult course. I think you just have to accept the fact that it's required and get through it. The content was much more interesting for me after the first midterm. Going to section will help you a lot.
If you don't have to take it for the concentration I don't see any reason why you would. If you're a CS concentrator, one thing to know — it seems to be much, much harder in 2019 fall than in previous years, judging off the time commitment and comparing the past exams to the 2019 fall exams. Proof writing is the main thing you need for this course, but it can be learned as you go, so you don't really need to bother with CS20 unless you want to. It won't help you that much, and CS121's Homework 0 will be a good, low-stakes way to develop your proof writing skills. Boaz has some good lectures and some bad ones. The textbook is valuable in some places but also includes a lot of information that's way too in depth for the course.
How well you do in this course is a function of how much time you spend on it. The grading rubric is ruthless and there is no error carried forward. The content is genuinely interesting, in terms of knowledge gained I have gained a lot, but grades are how well you read the mind of the grader not how well you demonstrated knowledge. If the section/OH/Pset mix is refined this has the potential to be an excellent course. But instead I have never tried so hard in a course only to do so badly.
The psets could be a lot of fun, if you started early enough and weren't stressed. There are bountiful bonus points, which really does help with stress. Also, get a pset buddy and you'll be just fine.
Some of the grading can be a little inconsistent, but they do their level-best and there is a regrade option that is pretty easy to use.
It is so hard, especially for someone without mathematical background. You will die because no resources exist for students. I wish there was more being done for students, but instead, the exams feel like they are catered to the select few that understand while most other people will die. It's also super frustrating to study so much only to score so so low; it's demoralizing and has made me reconsider the field of computer science as a whole.
This is a good treatment of the theoretical underpinnings of computer science, providing good mathematical intuition in clarifying some of the most exciting problems in computer science such as P vs. NP, defining computational models, and randomized algorithms. My few gripes with this course are that some of the teaching assistants themselves did not seem to understand the material that well, as it often was the case that they would recite the answer solutions during office hours (this is not to say all TA's, only the majority of them that I have attended office hours for). Moreover, when it came to grading the problem sets and exams, it seems like a lot of the times they would pattern match your solutions with the answer key and if your solutions did not have some buzz words found in the answer key they were using, you will most likely receive no points on your pset/exam. As a result, I had to spend a lot of time creating regrade requests, to which I got around a 90 percent success rate, a waste of time for me and the TA's had the TA's just read and reasoned about our homework solutions correctly in the first place.
You will obviously have to take this course if you are concentrating in CS, so there's not much you could do there if you're thinking of taking this class. Therefore, I will make a few comments on the decision to concentrate in CS. Most people come to Harvard to concentrate in CS because it offers the best job prospects after graduation. This class will not necessarily give your skills to don on a software engineering resume, but it will teach you think in a different way. In this way, changing your mindset from "getting a job" to "learning for learning's sake" will serve you much further in this class and onwards as you continue on your studies down the road.
This class was really good. It was a good introduction to theoretical computer science, and it covered a large range of topics, which made it interesting. It was not too stressful, yet I feel that I learned the content quite well.
This class was super stressful for me. The lectures were super confusing as well as the textbook. So the last chance within the

Comments
<p>course to learn the material was in section, which sometimes was useful but most of the time the TFs got tripped up with terminology as well. This class did not cater at all to students without an extensive proof-based math background. The terminology shown in lectures and in the textbook was super confusing and material was not well-explained. Section notes were super useful though, and I gained most of my knowledge using those and outside sources. Also the midterms were unnecessarily stressful in that the difficulty and number of questions was not fitting for the time allotted to us. The TFs were great and I feel like the professor did really care for students' well-being, but I felt like he was somewhat out-of-touch to the actual feelings and struggles of the students. However, I did feel like the struggle of the course definitely helped me grow as a student and eventually I felt like I learned the material towards the very end when studying for the final. During the semester though, the experience was pretty awful.</p>
<p>This course does a good job of teaching you. However, if you already have experience with proofs, it can be a bit ridiculous. The course conflates rigor with excessive notation. The graders have very high variance, especially at the start of the courses.</p>
<p>Boaz gets a really bad rep for some reason. He really shouldn't. Don't take the MIT class, it is not worth it. Just take 121.</p>
<p>This class is essential if you want to understand what the possibilities and limits of computing are</p>
<p>This course is difficult, but definitely manageable. Office hours are really helpful for the psets, and I wish I had gone to them more often. Like most CS classes, this class is pretty time consuming, but it's not ridiculous and the course staff does a lot to try to make it manageable for everyone in the class. You'll learn a lot in this class</p>
<p>Take it if you have to for CS, it's not the worst but it's also not for everyone. If you're not a CS concentrator you most likely at the VERY LEAST want a secondary, because otherwise you probably either don't have interest in the material or would be unprepared for the rigor.</p>
<p>There have been some negative reviews of the course in the past, leading some students to take a similar course at MIT. I think that these negative reviews were somewhat unfounded and may have been the result of the difficulty of a first theoretical computer science course more than having to do with the course itself. I found the course useful and Boaz seems like he really cares about student success and mental health.</p>
<p>Professor Barak is a great guy. He's really trying his best and is a very accessible, kind teacher. That being said, CS 121 is a messily taught class. It's not awful— the material is important and interesting. It just doesn't convey any of that well at all. Yes, you eventually learn (some?) of the material while cramming for tests and begging the one of the rare TFs/CAs who can actually explain things coherently for help. But lectures are a mess, the textbooks are a mess, and the course staff is an utter train wreck. Office hours are pointless, don't ever go— just read the (unfinished) textbook. Course staff is often belligerent and defensive. PSets will not help you learn much that isn't already intuitive from the textbook; a lot of it is sheer busy work. If you have the time to commute to MIT, I urge you to consider the MIT alternative because I'm kicking myself for not doing that in the first place.</p>
<p>Honestly pretty reasonable if you put in the time and effort (wish I did more...) — there is a lot of support from the teaching staff, and the textbook/lectures are useful for your needs. Go to/watch lecture, go to section, and go to some section and you are golden. It can be difficult to stay on top of things if you ever get behind though :(</p>
<p>Personally, I loved this class. However, I preface this as someone who enjoys theoretical mathematics more than programming. I found a lot of the material quite difficult and unintuitive, and went over concepts many times before understanding them. Be prepared to work hard and spend a lot of time on problem sets, especially the later ones.</p>
<p>There are 10 types of people in this class: those who like theory, and those who don't. If you like theory (and you may have to discover that you like theory in the process) you'll get a lot out of this class and learn some really interesting stuff. If you don't, this might be tough. Psets can get very difficult and collaboration is imperative.</p>
<p>Make sure to go to office hours if you're ever unsure about anything, because there will be parts of the textbook or topics that are inconsistent with what we've previously learned. Theory gets fun though once you commit a lot of time to it</p>
<p>I put everything I had into this class and still did badly. So discouraging.</p>
<p>The problem sets are time consuming and difficult; definitely get a peer group to collaborate and go to office hours. Also practice writing reductions, since it will make a difference on time sensitive tasks, like exams.</p>
<p>If you're not CS, don't take this class. Seriously. If you are, it totally sucks, but you don't have a choice. The pros are you definitely cover a lot of material and there's a lot of office hours. Almost everything else is a con. While Boaz cares about student well-being even if he doesn't really know how to deal with student stress and panic and clearly knows a lot, he's not a good lecturer. Lectures moved way too fast and concepts were not explained thoroughly – I was often lost after 10 minutes and couldn't recover. I learned most things from the textbook, which I had to read multiple times over to extract information from. The TFs often know the subject matter but are rarely good at conveying information in a way that is not scary and stress-inducing. The problem sets were so unreasonably difficult and painful that I had no hope of answering a single question by myself, even if I were to think about it for hours (which I tried). They were way harder than exams, and the subject matter often bore very little resemblance to what we learned in lecture or in the textbook. The exams, although still very difficult, were at least fair. This class made me feel beyond incompetent and decreased what little interest I had in theoretical computer science to zero. I am very unencouraged to continue in the computer science concentration but I have resolved myself to having a painful experience.</p>
<p>This is an easy class. You just have to spend time on it. There is a big difference between hard and time consuming. This is a time</p>

Comments
consuming class, but it is easy once you spend the time. Go to office hours! Do practice tests, and have fun.
Most of you have to take this course anyway so it doesn't matter, but this course is horrible. Be sure to take advantage of OH long before the psets are due. I found the textbook to be more helpful than the lecture, but that isn't saying much because the textbook is incredibly difficult to parse and contains tons of gaps. The most useful component of this course is probably section.
This was a hard class, but I did end up learning a lot and the content was pretty cool. Professor Boaz also cares about the course and students, and it truly makes a difference.
You have to take this class because it is a requirement, but honestly is the least applicable class every and I don't think I will ever use anything that was taught in this class. Boaz is a nice guy, but terrible lecturer. Even if I spent hours and hours studying for the exam, I still did terribly. Problem sets were also very difficult. I put in a lot of work, but it was so frustrating because it seemed I got no return from it.
The progression of topics that you cover is very well organized and thought out. However, at times the class can be tedious and disorganized in terms of the assignments and expectations.
This class may not be particularly useful unless you want to go into theoretical computer science (unlike a CS124), but the material is interesting. The psets can be hard to understand at times, but since everything at worst has 20% extra credit the class isn't too difficult. Put in the time and you will do well.
this class is fine if you go to all the lectures, sections, read the textbook, and study for the exam
I would not take this course unless it is required for your concentration. If you do have to take it, I would definitely recommend taking it with friends. Overall, it's a fine CS course, and Boaz is very friendly.
Section notes are extremely usefull and going to office hours.
You need to be pretty good at doing proofs and also *reading math language* so if you are not EXTREMELY confident TAKE CS20 IT WILL HELP YOU. Go to section because that's where you will learn the most (textbook and lecture are unclear). also MAKE FRIENDS TO DO THE PSETS with because i was by myself and got destroyed on them which was not good
The class is decent overall, and not too difficult. Problem sets can be really easy or really hard. Same thing with midterms. The TFs are generally super helpful. The textbook isn't great, and lectures weren't too much more help.
The textbook, psets, and notes often have typos, so be careful. It will be confusing and painful at times, but there are more opportunities for extra credit than you'd think, so it ends up being fine. Make use of office hours with TFs who actually know what they're doing.
Very interesting material that is not the most well taught. Boaz is a cool dude but there are a lot of weird things with grading like both positive points and negative points, and also a lot of extra credit points to offset the lack of curve.
I am so glad that I'm done with this class. I'm sorry if you have to take this. If you're concentrating in CS, you have to take this or the grad version of this class at MIT (MIT might be a better move since it seems more organized and more applicable... aka no focus on NAND) so you're out of luck. If you're not concentrating in CS, I wouldn't recommend it unless you're really interested in cs theory. Find a good group to pset with and try to find a good TF. Good luck!
Professor Barak seems to have made a huge effort to be more accessible to students and has done so successfully, in my opinion. Although the textbook is not yet in its final form, I found it very helpful to read. The lectures follow the textbook quite closely.
Make sure you stay on top of problem sets, definitely go to office hours and collaborate with peers to brainstorm ideas for solutions. Sections and reading the textbook are all really good ways to crystallize the material introduced in lecture. This class is definitely tough but manageable. I didn't take CS20 or any pure math class before taking 121, and found it to be perfectly fine.
Tough class you have to take. Important to get into a good routine of going to office hours and meeting people in your class. If you don't have proof-based math knowledge, definitely take CS20! (Ideally do more)
This is one of the worst classes in the CS department. Leave it till your senior year so that it won't make you want to switch concentrations. Way too much work for no real benefit unless you love theoretical math for its own sake. This course made me hate theoretical CS.
Teaching staff is robust and class is improving. Not as bad as the 2018 ratings of ~3.3 make it seem.
On the first day, Boaz explains the poor Q scores by the equation that "Required + Hard = Sucks." I was worried, but I ended up really enjoying the class.
Here's some advice: <ol style="list-style-type: none"> 1. Read the textbook chapter BEFORE lecture. It might take 1.5–2 hours to understand it deeply. The time investment is worth it, because you will learn the material so well that you probably won't have to go to lecture. 2. Start the homework early. Most of the psets are solvable in 1–2 sessions of office hours. If you finish it before others, you'll have a chance to explain it, which (in my experience) helps you understand it better. 3. Work on speed for the midterms. If your second midterm is anything like ours, you'll be under a lot of time pressure. 4. Don't stress about your grade! The psets all have at least 30 points of available bonus that carries over to other psets, and the midterms / final have upwards of 20 points of bonus each. You will be fine.

Comments
If you stay on top of the material and do things early, you'll do really well in this class. I took Math 23A/C and that was enough proof background. Shoutout to Boaz, Madhu, and the TFs — they all really care about the students and are great people :)
Get a pset group and try to be on top of things. Some material in the course is challenging but don't be discouraged. You might not need to know it.
From what I've heard about the class in past years, Boaz has done a good job adapting the course in response to its reputation as one of the more challenging CS courses. If you're a CS concentrator (or considering a concentration), you definitely shouldn't shy away from this course because you will learn a lot of foundational CS theory and there is a lot of support and cushion built into the course. On the other hand, if you're not in the CS department, I wouldn't recommend this course as a CS elective because the material is a little dry. (I would recommend 124 instead if you're interested in theory, and you definitely don't need 121 as prep for 124 if you're truly interested in the content.)
Boaz is a great lecturer. He tries really hard to help everyone understand by encouraging question asking and going over things carefully. I feel like Madhu only called on the section kids (all male) but then again there's a small sample size, he lectured like twice.
I think people would enjoy lectures more if they really paid attention (speaking as someone who often sat in the back with a computer, I learned and had much more fun when paying attention up in the front).
There is a ridiculous amount of extra credit, but I think it is actually quite effective at its purpose of reducing stress without reducing learning.
Go to CS 121.5, there's really interesting side topics and the food is good :)
Boaz is a great man but not so a great teacher. Still, you can tell he tries a lot and cares a lot about the students and the subject. When you take this class, please clap at the end of the lecture. Thanks.
Be prepared, form study groups, and try not to stress yourself out too much. This stuff is hard, but do-able. You got this
homework is painful and tedious. Imagine spending a long time figuring out the solution (don't even think about working alone, work with others), and then realizing that you have to spending another 1–2 pages to type up the tedious solution about induction or the classic 4 step reduction problems. And then having to do this for 8+ more problems. (for the worst psets, the solutions add up to 15 pages of latex!) The material is interesting but the homework and lectures are painful and hard. The class is pretty disorganized – TFs not knowing how to do the really hard and obscure questions without the answer key and having to message on the slack for constant clarification for poorly written problems. Combine this with hard material and its understandable why a lot of us have felt despair preparing for the midterm and final. Even though its painful I don't regret taking this class, I've learned a lot – just be aware of what you are getting yourself into.
Get ready to be confused and sad a lot.
This class is very difficult. To do well make sure you have a good pset group.
Go to lecture, no matter how nonsensical it seems. Go to as many office hours as you can but come prepared with questions for the TFs. Start the problem sets at least 3 days early if not a week. Go to section and make it part of your weekly routine.
Take this course. It's very difficult, and you will struggle, but you come out of it having learned a lot. Office hours and section are extremely helpful. The textbook is hard to decipher on the first read, but it starts making a lot more sense when you're reading a particular chapter for the third time (which I highly recommend you do). It is very easy to fall behind on this course, so make sure you actually do the readings and attend section — they'll help A LOT with problem sets and exams. The problem sets are hard and long, but office hours are a great way to get them done and the TFs are fantastic. The exams are extremely reasonable, and as long as you have done the problem sets, you will find them doable. Ultimately, you end up spending a lot of time on this course, and completing the quizzes, problem sets, readings, and exams is a struggle; but it is not hard to get full points on each of those components (there are tons of bonus points), so your final grade ends up being very fair, mores than other CS classes. This class also had an extremely supportive environment — Boaz is super kind and understanding, and really cares about this class and his students. I spent a lot of time on this class, and really had to grind to keep up, but it was worth it, and I have definitely learnt a lot of material that I would never have otherwise been introduced to. Take it.
CS121 takes a lot of work, and the material is very dense and difficult. Expect to spend upwards of 12 hours per problem set and spending a lot of time at office hours. Boaz tries his best to teach the class, and truly cares a lot about learning rather than grades, but his lectures can be quite hard to follow and it's easy to get confused quickly. Reading the textbook will help a lot, and use Ed whenever you have questions, Boaz and the TF staff are very responsive.
Sections are extremely helpful to reinforce content from lectures, especially because the latter can be kind of confusing at times. Don't be afraid to ask the TFs and instructors questions — they're more than glad to help!
Don't take it. I was considering doing the MIT equivalent but was lured in by the promised changes to the class. They were insufficient. This is the kind of class where the professor assigns a problem set due over Thanksgiving break but cancels office hours (all of which are, of course, needed to complete the impossible homework) over break. One might consider that, if the TFs shouldn't have to work, perhaps neither should the students. This has by far been the most unpleasant class I have taken at Harvard. If the problem sets and exams were removed, it would rather quickly become a genuinely interesting class.

Comments
<p>It's better than I thought it would be, honestly. Boaz is a pretty endearing instructor, he cares a lot and has fun slides. The assignments can be absurdly tedious: one pset had us writing 3 page long proofs per problem. Thankfully, they've loosened up the collaboration policy. 2 out of 3 exams were fair, midterm 2 was far too time-crunched. Overall, this is a decent class but you will do best if you focus on the important material and proof strategies in it: reductions, computability proofs, etc. There is so much material that you need to just focus on the important stuff, frankly the long and incomprehensible proofs in the textbook can be skipped as long as you understand the top 5 basic proof strategies.</p>
<p>If you are uncomfortable with basic mathematical formulae and symbols, this might be a bit of a struggle, otherwise the course is largely well structured.</p>
<p>This class is infamous but I enjoyed it to an extent – the material, while not super relevant to the software engineering jobs many CS concentrators hope to obtain after graduation, was interesting and helpful to provide a more wholistic understanding of computer science, specifically the units on complexity and randomness. Assignments sometimes felt like busy work with a lot of writing, compared especially to CS124 which I felt required more thinking. Midterms were fair but time was rushed, but the final was very reasonable and there were lots of bonus points provided at every option – take advantage of these as the class is not curved!</p>
<p>I will preface this by saying that Boaz is an extremely nice professor who really cares about students.</p> <p>However, do yourself a favor and take the MIT version of this course. Boaz claims that 121 is a "more modern" theory course, but after talking to friends who took the MIT Theory of Computation Course (and loved it), it seems like the two courses cover very similar concepts, but 121 also highlights uniform computation – aka computation using circuits, which is somewhat interesting but not the most essential concept to focus on for weeks. So, 121 essentially just does a less effective job at teaching the material.</p> <p>This course just isn't very well-run, motivating, or engaging at all.</p>
<p>This class is great if you get it and terrible if not. Unfortunately you have to take this course as a CS concentrator. Good luck, get a peer tutor, and work with others.</p>
<p>Class is a lot better than what it could be. Boaz cares a lot about making this class great and a lot of parts are. The textbook is excellent. Lectures are also fairly informative but also just reiterating the textbook. This class did make me more deeply interested in CS so I would def recommend it. The material is genuinely fascinating.</p>
<p>Very difficult course. Arguably a little too difficult for its purpose (*introduction* to theoretical CS). HOWEVER, the teaching staff genuinely care, which makes this course superb. This class consistently reminds you that you have a ton of support and that GROWTH matters more than getting it right on the first try. Office hours, sections, lectures, teaching staff (Will is an incredible TF), and Boaz himself — there are many resources in this class, so you will never feel alone. I grew a lot from this course and despite its difficulty, really enjoyed it. It was more proof/math-based than coding-based, which I loved. Psets are very very doable, just start as soon as you can and go to OH. Exams are tough, but you'll be okay. The class isn't curved, but I think I'm more proud of the growth I experienced and am not really expecting much from the letter grade. I have a newfound hope for CS!</p>
<p>Most challenging course I've taken at Harvard. Make sure you have formal math background or else you will hurt. On a more positive note, the material is incredibly fascinating, and the teaching staff are trying their best to make the course better. The teaching staff are also incredibly accessible, and Boaz clearly loves teaching and talking to students. If you are curious, patient, and keen to learn then you will do well in the course.</p>
<p>It's a shame that a class that is required for CS undergrads is this poorly taught and organized. Unfortunately, you have to take this if you want to study CS, so despite it being AWFUL (inane psets with inconsistent support from TFs, horribly taught lectures, half finished textbook and review materials provided to students), I'd say it's worth suffering through to get to the better classes. Don't be afraid to submit stuff for regrades (grading is very inconsistent so you'll often get many points back) and email Boaz with concrete suggestions of how things should be changed, as he's relatively open to feedback.</p>
<p>Amazing course! A thoroughly interesting and engaging introduction into theoretical computer science. The material is super interesting and relevant to so many other areas of CS. PSets are approx. biweekly and the problems help reinforce the subject very well.</p>
<p>You may hear that the lectures for this course aren't the best, and this is relatively true. The lectures don't always cover all the content in the book and can move somewhat slowly. However, this shouldn't prevent you from taking this course. The course material is interesting, the textbook is thorough and instructive, and most of all, Boaz is an incredible person. If you ask a question, he'll unfailingly commend you for asking a great question and answer it enthusiastically. He puts a lot of effort into making this class accessible and enjoyable for all his students. On top of this, the assessments and problem sets were extremely reasonable, and the grading was also quite generous, especially with all the bonus points. Note that if you have a strong math background, you'll likely find this class to be very manageable.</p>
<p>This is a very interesting class. I think its a rare opportunity to learn about the theory behind certain kinds of problems. Would recommend!</p>
<p>This is a wonderful class, but please please please 1. Go to section 2. Find a good group of pset buddies 3. Stalk Ed and 4. Go to office hours. This class can be pretty difficult.</p>
<p>Some tips for CS121:</p>

Comments
<p>–Get ready to be confused for a large part of lectures, problem sets, and exams. This isn't necessarily a bad thing, as it can force you to do a lot of independent learning, but you most likely will not learn everything you need through lectures and the book.</p> <p>–The problem sets vary in length and difficulty from week to week, so start them early. There's really no way to predict how long a problem set will take, so don't leave them for the last day.</p> <p>–Take the class with friends. Office hours are very rarely helpful, and unless you have an EXTREMELY strong math background, you will not be able to complete the problem sets without a small hint to get you on the right track here and there, so discussing problems with friends is a huge help.</p> <p>–Understand your answers. This is mentioned in the collaboration policy, but it is super important to really understand the proofs you write for later in the class, so don't just put down bits and pieces of what you hear from your friends.</p>
Don't take this if you don't have to lol.
<p>If you have competitive programming experience, this will be a course that is a little bit hard and time-consuming, but not as difficult as people say.</p> <p>In high school, I took a semester-long course on algorithms/theoretical CS (so I did reductions before) and spent on average 13 hours a week outside of lectures (I used time tracking software). These hours even include studying for finals and some of the optional CS 121.5 sections, so in reality you may spend less time.</p> <p>There are a lot of bonus problems in problem sets and you only have to submit a problem set every 1.5 weeks. The exams also have a lot of bonus problems (our 2019 final had a total of 133 points) so you do not have to worry about making small mistakes here there.</p> <p>One thing I regret is to not go to sections more. Even though sections are optional, they give you some extra stuff that you can use in problem sets and solidify knowledge.</p> <p>This class is very rigorous and interesting, but it is poorly organized and really awful. If you can avoid it, do so; I would rather suffer through 124 without 121 than take this class. The homeworks are way too long and involved, so even though you usually have 10 days to do them, it's only by the last 3 that you actually understand enough to figure them out.</p>
Make sure to take advantage of office hours. You can't do the problem sets alone.
Easy class, tough exams. Lectures are a waste of time as they are nearly incomprehensible. Read the textbook and you will be fine.
The material can feel very theory-heavy and un motivating but find yourself a good support/study group and go to office hours! The TFs and course staff are always there to help
<p>Problem sets were well designed and engaging.</p> <p>Tests were reasonable in difficulty.</p> <p>The textbook is super thorough and a great resource.</p> <p>Lots of care was put into this course.</p> <p>The material itself is quite dry, but I really do think the course made it about as palatable as theoretical computation can be.</p>
Boaz and the teaching staff for cs121 are incredible and put an enormous amount of time into supporting the students. That being said, the textbook is pretty difficult to understand so going to section and office hours is kind of a must if you really want to understand the material. But don't let others shy you away from taking the class because it's not "applicable" because the theory is really interesting and critical to know as a software engineer.
It's required for CS concentrators, but there is at least some good news about the course. It is much, much easier than CS124 and Boaz is a good lecturer, which will help get you through some of the more complex or boring parts of the material. The in-progress textbook still needs some work, but it's free, so you cannot complain too much.
Try to go to lectures despite them being optional. Boaz is really eager to help, but you have to ask for it.
Honestly this course was a lot better than the reputation it has. Sure, the material is very theoretical and not all of it seems interesting or important, but there's a lot of fundamental stuff here when it comes to understanding the types of problems we can solve with computers. Moreover, Boaz is a really dedicated instructor who knows the material better than anybody, making him a great resource. The psets kind of felt like a grind toward the end of the semester (they seemed more frequent and difficult), however this might be just down to end-of-semester fatigue. All in all, this is not the class to fear as a CS major, so try to have a good attitude about it!
Not as bad as people make it to be. Lots of bonus points to make up for missed points. If you manage your time well it is very doable along with another hard class.
This class is pretty hard but there are a lot of cool topics covered. Take it with friends if you're a CS concentrator or if you like math. There's a lot of math and proofs so be ready for that. The textbook is really helpful as an extra resource if you didn't understand something from lecture.
I would suggest taking this class in sophomore year but going in with a group of friends to have active groups you can work with and follow the course along. It's time consuming so try to have no other hard classes.
Make time for this class. Don't take it during a difficult/swamped semester. Pay attention to the online discussion forum, go to

Comments
section if you can, attempt all the bonus problems on the homework, and find a friend or group of friends to brainstorm together for the homeworks. That should make the experience easier.
I came in thinking I wouldn't particularly love it, but it turned out to be a great class!
Eh it's fine. I took as a senior after taking 124 and I don't recommend doing it that way because it was kind of too easy and not super engaging but also all the sophomores seemed really stressed. The material is not super interesting but Boaz tries. There is plenty of chances to do well if you are motivated, it is more effort based than some classes with the bonus point homework policy.
First of all, if you are not a CS concentrator, do NOT take this class. If you are CS and have to take this class, good luck. I'll start with the textbook, which is woefully inadequate. Boaz has insisted on teaching this class from his textbook "in preparation" (read "not finished"). The textbook is only available online, and does not resemble anything close to a finished product. Formatting issues, crudely hand-drawn figures and the frequent use of TODO make this book very difficult to decipher, as does Boaz's tendency to explain concepts in the most complicated way possible. This tendency extends to lecture, which is confusing at best. Section is very helpful, as are office hours. Psets will definitely take time but are doable, especially if you attend office hours and section. The course staff includes some very dedicated and excellent TF's (Will, Madhu) who can help you find your way in this poorly conceived and horribly executed course. My best piece of advice would be to find and utilize any and all outside resources you can, because you will receive little help from Boaz in understanding the material.
Solid course – I think anyone with a decent math background will have no problem and I think anyone could do well if they put in the effort and use the (many) help resources offered by the course. The material was dry at times but overall was enjoyable
Don't listen to the haters. This class will not kill you. If you put in the effort, the course is very manageable. The teaching staff is very helpful and wants you to do well. CS20 can be very helpful to get you used to writing proofs as well as introduce you to some of the foundational topics for the course, but I wouldn't say it's an essential prerequisite. Overall, this course teaches you a lot about how to think about algorithms and how to transform seemingly unrelated problems into one another.
This course is really badly taught. But, if you're taking it you probably don't have much of a choice anyway so buckle up I guess! If you can, take the MIT version. The textbook is much better and the psets are the same difficulty. Also, definitely don't take this course unless you have no other option. I get it that theoretical computer science is really interesting, but there are plenty of other interesting courses that won't be as frustrating.
The course staff definitely puts a lot of effort into making the class as good as possible, but the class is mostly difficult and uninteresting. While the subject matter is hard to teach in an interesting way, many improvements could be made to the textbook, lectures, and design of the psets.
This class is very very difficult and requires a lot of independent study to not fall behind, along with relying on other students and office hours in order to complete problem sets. However, you will learn quite a bit and even though Boaz knows the class is hard and makes hard assignments, he cares about the understanding and mental health of his students (their grades, not as much)
Professor Barak is a great person. The material can be very hard and confusing, plus the course moves very fast. Start psets super early, they always take longer than expected and eventually you will run out of late days. Office hours did not feel productive for the most part, except for the office hours of this one specific TF, who truly made it feel like I could ask any question without being judged.
The class might be challenging sometimes, but the professor is really trying to make it as good as possible. The material is important and even though it might seem irrelevant it helps understand many topics beyond 121. Psets are doable especially if you go to office hours. Highly recommended!
CS 121 was a very fulfilling course that teaches key concepts, such as computability and efficiency, as well as problem-solving strategies. Boaz puts a lot of effort into improving the course each year. He also really cares about students and emphasizes learning; this is evident in the fact that there are extensive opportunities to earn bonus points (on exams and PSETs, as well as through optional projects and lecture/section attendance), and the late days make PSETs more manageable. The textbook is extremely detailed and comprehensive – even more so than the lectures. Overall, CS 121 is an extremely well-run course and one of the best I've taken at Harvard.

What did you take away from your experience in this course? What did you learn? How did this course change you?

Comments
I learned that HALT is uncomputable, and so is the rudeness of the staff.
I learnt a lot of theoretical computer science and I feel much more engaged with the department.
I learned the theory behind what computation is and what it means to compute a problem.
I took away a deep understanding of the powers of computation in the modern world, and the scope of progress in the area.
Very, very little. This class nearly crushed my desire to continue with the concentration.
I got a much stronger foundation in CS theory.
I learned about the beauty of math in CS.

Comments
Learn to think about some of the theoretical math required of CS, some of which I had never considered, such as the types of programs that are programmable, as well as what makes an efficient program.
I feel like I got a great introduction to some of the most important topics in theoretical computer science.
I took away that I don't like CS121 at Harvard. I already took 124!! Why should I have to take this class! I learned 0 interesting concepts.
I know much more about computability, complexity theory, algorithms, and more. It was incredibly interesting and informative and I would take it again given the chance.
I went a bit deeper into topics like computability and space complexity than I had before.
I've always been curious about Theoretical CS, so this course has been a good introduction to the field. I have definitely learnt many concepts from this course, and will be interested in exploring some of them deeper if I have the opportunity to.
Learnt I wanted to computer science
that even though i found theory interesting at first, i shouldn't pursue it
Computability and stuff, I guess.
Did learn some interesting properties of computability but will not use it in my career path
Better prepared for higher-level CS
This was a good class to introduce the ideas of theoretical CS (and discreet math, as I had never done it before). I learned about an entire new way to think of problems.
I was able to understand from an abstract standpoint the key concepts of computer science and truly understand things that I took as mere truth within the field, such as runtime and efficiency. I learned the power of proving concepts and how this power can carry to abstract and seemingly unknown problems.
I gained an appreciation for theoretical CS, which I had no exposure to before. I got more comfortable in and confident with CS throughout this class. And, honestly, the difficulty of this course — which was greater than my freshman year classes but not insurmountable — actually helped improve my time management and studying skills a lot.
Very comfortable with the basics of CS.
Although it was the hardest course I have ever taken, I think I learned a lot, and am especially thankful for the exposure to proof writing techniques.
I am incredibly frustrated with this course. I really studied hard, but every midterm/final I walked out thinking I did well and got destroyed in them because of a small error – it baffles me how a single error in the working of a question warrants losing all the available points, even if general understanding is there.
I learned a lot about theoretical computer science, which isn't something I knew anything about before hand.
I will never take such an intensive math/cs class again
I learned a lot about theoretical computer science. I also became a better problem solver.
Experience writing sound proofs, basic theoretical CS concepts
Became much more interested in theoretical computer science
I was hoping I would take away a good foundation in theoretical CS, and I feel like I got what I was hoping to get.
Actually learned a lot and viewed computation differently. I think the course actually changed me in a good way, but it was tough to get through
I realized the power and applicability of theoretical computer science to solving real-world problems or even to show that problems are not possible to solve.
I learned to never go to office hours for a CS class again. I also learned how to read proof outlines and CS-style "exercises left to the reader".
Should manage my time better and do more practice / put more time into understanding it conceptually. Pretty interesting to be able to be pushed to think computationally, and got excited about some of the extended topics (like cryptography).
I learned that I enjoy theoretical CS more than programming.
Everything is a string and non-trivial semantic functions are uncomputable. Also $P=NP$ (or not)
I became more interested in theoretical computer science and am thinking about more related courses in the future
I learned some things about theoretical computer science but also that academic computer science is definitely not for me and that CS is very, very difficult.
I can now make jokes involving zero functions and computability. I learned about P vs NP. I now understand the importance of theory and am thankful for what I have learned .
I learned how to deal with a great deal of stress thanks to this course. It made me a much more resourceful and independent

Comments
learner because many of the resources provided by the course are not helpful.
This course traumatized me
It was good to learn a lot about different proof techniques.
difficult but could be rewarding
I learned how to approach really difficult concepts
I feel so much more confident in my mathematical abilities after taking this course. When I enrolled in this course, I was scared of it and I thought that I hated theoretical computer science. Now, I am inspired to take more theoretical computer science courses. I might like to take Professor Barak's course on cryptography.
I learned different proof formats, and gained a good introduction to theoretical computer science. There were many interesting topics introduced that I enjoyed learning about.
Found the course very interesting! Cool to learn about Computer Science from a completely different angle from everything you've probably experienced.
This course made me hate theoretical CS
First theoretical CS class. Theory is pretty interesting.
I am now more interested in theoretical CS and learning about the limits of computation.
I really loved the big ideas in this course eg. What would happen if $P=NP$
I'm more confident that I belong in the CS department because I feel like I succeeded in this course without much strain. However, I do feel like I should challenge myself a little more in the future. The course has also sparked my interest for topics like crypto and quantum computing.
A better grasp of the content and how much I can push myself in the field of computing
uncomputability, halting, programming languages, universality the cool ideas about CS theory
I have a much better understanding of many CS concepts, including P, NP, P vs. NP, BPP, etc.
Learned how to write proofs and learned a lot about P and NP
I feel like I've learned a great deal about computer science from the theoretical side of things, which definitely excites me and motivates me to continue studying more TCS!
I learned about how we define computability, broadly, and also interesting problems in the world of computer science.
Learned more about the foundations and the math behind computer science!
Gained some more experience with proofs.
Made me glad to be done with the course and ready to move on to other things.
I learned I want to do some research in computer science.
I am able to think much more logically about problems, both math-related and not.
This course definitely changed me as a person. It taught me to approach problems in a completely different manner and to be patient when approaching a challenge. It taught me that theoretical computer science is incredibly beautiful (albeit extremely difficult). Honestly there is so much more but it is hard to list here!
I learned to appreciate theoretical CS much more — it's a fascinating subject with so many interesting sub-areas (e.g. time complexity, space complexity, quantum, etc.) I now have a much deeper interest in pursuing this type of material.
This course sparked my interest in theoretical computer science, especially in the area of questions related to the P vs NP conjecture.
Learnt that different kinds of problems have provably different levels of difficulty
I remembered why I started liking computer science in the first place. I became closer to a lot of good friends too. The best friends are the ones with whom you survive trial by fire, I guess.
I learned that theoretical computer science is not the field for me. I am, however, glad that I took the class because that's a really important thing to figure out, and I'm glad I tried it.
A little better at writing proofs, I guess.
Machines can do the same things as brains can do, albeit brains can probably do some things more efficiently and vice versa.
I took away an experience with a truly challenging course. Additionally, I learned some tenets of theoretical computer science which may prove useful.
From this course, I took away a high-level understanding of computability, poly versus exponential complexity classes, reductions, and Cook-Levin. We learned about how to have your cake and eat it too! 121 gave me an appreciation for theoretical computer science and understanding the importance of certain problems and the ramifications of solving them. This course definitely pushed me farther in my pursuit of computer science!

Comments
Learned a lot about theoretical CS and think my proof writing skills got better.
This course made me better understand basic concepts in theory of CS. I learned a lot about computability and modeling computation formally as well as efficient computation and complexity classes. Now I have a solid understanding of things like reductions and P vs NP.
I learned the foundations of theoretical computer science, specifically, different computing models, computational complexity and the relationships between different complexity classes.
This was a challenging course for me but mostly due to my own packed schedule. I still learned a lot about theoretical computer science and felt like I did alright on the course overall. The TFs and the professor want you to succeed, and if you find a good group of friends to work with, you'll be okay.
Transformed my perception of CS and made me realize that I love thinking about the theoretical aspects.
I learned that P may or may not be equal to BPP. Just kidding I learned more than that.
I learned some fundamental concepts about computer science
I had no idea what Theoretical Computer Science was before taking the course, and now I find it really interesting and am definitely interested in taking more TCS courses in the future.
I learned how to think like a computer scientist. I got more experience with proofs. This course did not change me as a person, except to diminish my desire to ever study theoretical computer science.
I gained some understanding of theoretical cs.
I learned a lot about the large picture of computation and gained a respect for all theoretical computer scientists
Aside from teaching me a lot about TF, this course taught me a lot about myself, time management and being better about seeking help
I learned a lot about theory of computer science and where it is used.
I gained an understanding of key computer science concepts, problems (i.e. P = NP?), and problem-solving strategies.

Instructor Comments

Please comment on this person's teaching. (Your response to this question may be published anonymously.)

Comments
Madhu's office hours were extremely helpful especially before midterms. He really drives some of the concepts home.
He cares. That's about all I can say. I'm sure he's a good professor, but in a class as miserable as this one, it hardly matters.
Boaz was a great lecturer and really invested in the class and his students. Truly an amazing guy and a great person to get to learn from as an undergrad in CS.
He is an excellent teacher. Good pacing, engaging lectures, and generous grading.
Boaz is not perfect, and sometimes his teaching has room to improve, but he really cares about the student's learning, and his effort and enthusiasm are contagious — what more could you ask for from a Harvard prof?
It is often hard to follow the professor during lecture, as he goes to fast at some points.
Boaz is super nice, and clearly cares a lot about this course. His slides are very good, and his book is great. I personally found his lectures very clear and entertaining, although some found them more difficult to follow.
Incompetent lecturer and educator.
Clearly a very smart man but should not be a teacher. He lacks any sort of necessary qualities needed to be an effective teacher (he is unorganized, unclear, not understanding, ambivalent to student's needs/opinions). He seems like a great person, but should not be a head teacher.
I like Boaz a lot. He teaches many topics well, some of them poorly.
Prof Boaz really cares about the class and has put a lot of effort into designing the whole course, from creating the textbook to creating new platforms for students. He welcomes feedback and responds to them sincerely, even modifying some of the course content and deadlines accordingly. He also has a great sense of humor and is definitely worth talking to outside of class – his passion for his work is very admirable too!
Lectures are getting better, he is passionate and cares about this course, can still make topics a bit clearer
Boaz: great person, not a great teacher, you can't just magically expect people to get things you don't ever explain in lecture
Boaz is pretty clear in his lectures, I liked them.

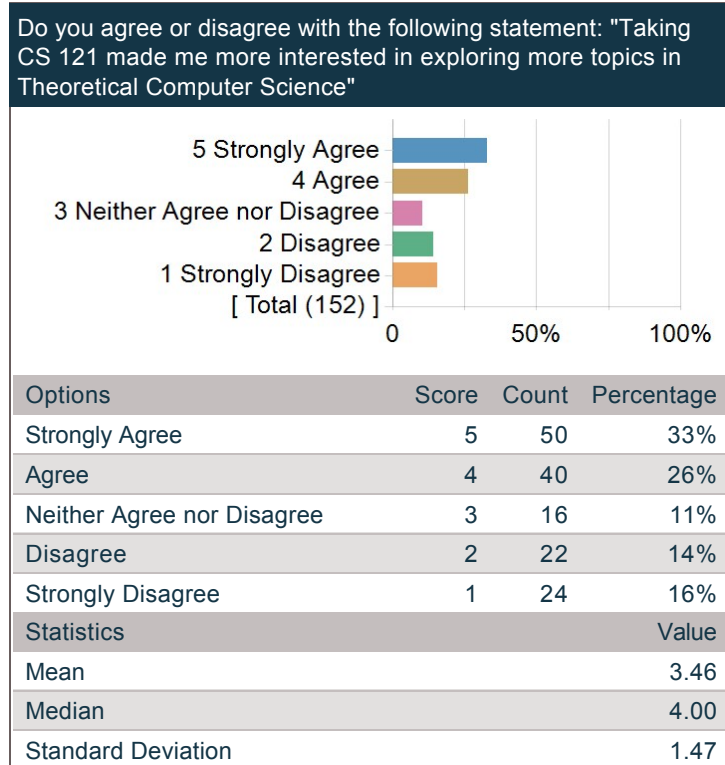
Comments
Boaz was great overall but his lectures were pretty dry and frequently more confusing than not.
Difficult to understand at points, I think he operates at a higher level that its difficult to understand notation and pass on intuition.
Boaz tries his best with explaining the material, which I greatly appreciate, but sometimes his lectures are hard to follow. There are times when it seems like he himself is confused about the topic, or he gets muddled in the way he is trying to explain something, which makes things very confusing for his students. I would appreciate more examples and more concrete applications to problems during lecture as that would help more for approaching problem sets and exams.
Boaz is a great lecturer; he outlines the major concepts and proofs of each chapter and illustrates how concepts come together. He's really responsive on Ed and dedicated to reducing student stress and helping students learn and succeed. I really appreciate how he constantly made tweaks to the class (adding a pset late day, increasing the number of cheat sheet pages allowed, releasing solutions as quickly as possible) to support students' health without sacrificing rigor of the course/learning.
Boaz is really intelligent. Lecture is not necessarily his strength but it is clear to see how passionate he is about the material. If you go up to him in office hours or after class, you will get a lot more out of the course. Also, he is extremely understanding when it comes to mental health issues/extenuating circumstances that might come up during the semester.
Boaz's enthusiasm and passion for theoretical CS is clearly evident and makes the course a lot more enjoyable. I find some of his lectures helpful and some not so much. I'd encourage him to hone in more on individual ideas during lectures instead of trying to give a rapid overview of all ideas, because in the latter case we don't learn anything as it all goes too fast.
Lecture stall word is still a big problem, most lectures and questions are directed towards a few students who keep up while the rest are often lost – if only 10% of the people in lecture are keeping up things might have to be changed.
Professor Barak does a very good job of supporting students and being invested in their learning. His lectures could use a bit of polishing though (sometimes there were random mistakes on lecture slides, or he'd run out of time to fully present a complex proof in class).
Boaz is so funny!
Super smart and super nice guy who wanted the best for us. His lectures were entertaining at times, but there's no doubt that it got confusing, and many lectures left the entire lecture hall thinking "what?". Regardless, he does his best and I think he should get credit for that.
Professor Barak is super accessible, really sweet, and genuinely seems to want people to do well and be interested in the class. That being said, his lecture style is confusing and not the most well-organized.
Boaz does try hard to make it fun and interesting, but at times he can lose most of the audience if it's an especially complex topic or he's running out of time
While Boaz cares about student well-being even if he doesn't really know how to deal with student stress and panic and clearly knows a lot, he's not a good lecturer. Lectures moved way too fast and concepts were not explained thoroughly – I was often lost after 10 minutes and couldn't recover.
The lectures are really good, though sometimes confusing. I learned a lot and am thankful for everything that you did!
Boaz is very nice and cares a lot about the course and the different topics. Sometimes his lectures can be a bit boring though.
tries to be engaging which is good
Could be a little clearer on some topics.
Boaz is super sweet and genuinely wants to help all of his students, but his lectures can sometimes be hard to follow. I think it's because he's so smart that he doesn't know what we don't understand.
Professor Boaz seems really passionate about the subject but lecture were hard to get through.
I believe that Professor Barak knows that CS 121 has a reputation for being a hard required class that some concentrators dread taking. I believe that he has made a real effort to make himself more accessible to students and his course more enjoyable to them. I think he has done a great job.
Lectures were a bit difficult to follow, but Boaz definitely cares a lot about the subject material and the students.
Boaz is a good person but a terrible teacher. His explanations and definitions are usually extremely hard to follow in lecture, and are only slightly better in his textbook. He does seem to genuinely care about the people he's teaching, which is good, but he does not do a good job helping students understand the material he's trying to teach.
Boaz tries to make his lectures funny which was really nice. He has nice presentations as well. Some lectures were too fast and I felt like I did not walk away with anything
Boaz isn't the most effective lecturer, but it's obvious he cares a lot about his students and the material he teaches, which counts for a lot. It's worth going to his lectures at least for the memes.
Boaz's lectures were good and lucid. There were sometimes mistakes on the slides which could throw people off. He was very encouraging during lectures wrt questions. Gender ratio of people he called on to answer questions was pretty good and he didn't do the cringy thing where he asked for more women to raise their hanhe; it was rather more of a natural thing.
I like how accessible he is and how he makes sure to remain active and answer questions on Ed – which is a lot more effort than a

Comments
lot of teachers put. And I appreciate that.
Clearly smart and knows a lot about the material. Needs to spend more time on actually making the course coherent though.
Boaz isn't the best lecturer, but he is a professor that cares a lot about the student's learning. He's also a really nice and chill guy.
Prof. Barak is super nice, funny, and knowledgeable about theoretical computer science! Although he sometimes has trouble explaining things, he is always willing to answer any questions and respond to help. Definitely a great resource!
Boaz is very enthusiastic but his lectures could be tightened up. Some of his proofs during lecture were long-winded and confusing, and those could definitely be improved.
Prof. Barak was clearly enthusiastic about the material and created a friendly atmosphere around this big class. His expectations were clear and he clearly wants his students to succeed. Sometimes his lectures are a bit confusing but I really appreciated his care for the class.
Boaz is wonderful! I think lectures could be a little more streamlined (sometimes there are slight errors in the presentation), but his enthusiasm is infectious and he is always willing to answer questions. I am really glad that CS121 is mandatory for CS concentrators because I think everyone should get the chance to learn from Boaz!
Great teacher that genuinely cares about students and teaching the material. I will say though that lectures can be a little more effective — specifically when teaching proofs (which were sometimes quite confusing).
Both a quick-witted and knowledgeable professor — he knows how to teach this material in a very effective way.
Boaz is one of the most encouraging and genuine professors I have met. In lectures, section, or anywhere else, if you ask a question, he'll unfailingly commend you for asking a great question and answer it with enthusiasm. He puts a lot of effort into his lectures and slides, and overall, making this class accessible and enjoyable for all his students.
I think it made a lot of students' lives unfairly challenging to have assignments due during thanksgiving and again during reading period, and that this should probably be eliminated. Aside from that, I very much enjoyed your delivery of this course, thank you!
Boaz has a great sense of humor and is generally a good lecturer, although he can at times lose the class when he goes too far into the specifics of one proof.
Boaz is a very funny man who is extremely passionate about CS121 and it shows through his lectures and his care for the students! Truly a game-changer.
Professor Barak does a great job of creating enthusiasm for the material and he really wants to ensure that everyone understands the material before moving on to the next topic.
Very enthusiastic and encourages participation in lecture a lot. Very approachable and open to questions.
Boaz is very responsive to email and you can always ask him questions. Lectures are engaging and mostly clear. His textbook is very well done and is a really good resource for the class.
Boaz clearly cares about the course and about his students. His lectures are interactive and the content is interesting. He is easily reachable either by email or by the online discussion forum. I enjoyed having him as my professor.
Professor Boaz obviously cares a lot about his students and it shows in his lectures!
Boaz is fine. He could definitely bring more energy. The lectures that incorporated videos/songs/cool pictures were more interesting. The lectures could be improved with more real-life examples, news stories, examples from movies etc that could just drum up a little more excitement. Having a song at the end of the lectures before the exams was fun and there is the occasional fun themed slide. More of this would be good.
Sometimes loses focus and clarity in lecture, and I think he has trouble with finding a happy medium between clarity and detail (going from surface level observations to long, tedious, and rather boring proofs quickly). However, he's clearly enthusiastic and encourages participation
Boaz is clearly enthusiastic about theoretical computer science and teaching. However, his subpar teaching made this course much more difficult than it needed to be. For example, he insists on using his textbook rather than Sipser's. I get that it's very hard to write a textbook and he should be proud of himself for singlehandedly writing an entire introduction to a subject, but his book is much worse than Sipser's, and undergraduates studying CS at Harvard have to suffer through it and finish the course without a solid foundation in theoretical CS for seemingly no reason because it is still used. Aside from the fact that every chapter contains several incorrect statements, uncompiled latex, and is riddled with typos, and the fact that a few key (testable!) chapters just straight up don't exist, the book is in really poor pedagogical form. Compared to Sipser, Boaz's book doesn't come nearly as close to tying each part of the chapter together into an understandable narrative. The heuristic examples that the book presents (for example the "have your cake and eat it too" thing is impossibly incoherent) actively cloud a student's understanding rather than helping it and many important theorems are presented in such a labyrinthine way that I wonder whether Boaz was competing in a CS Theorem obfuscation contest when he wrote it. The book always feels like a struggle to get through, and, although this isn't Boaz's fault, his English writing seems clunky and uninspired when compared to Sipser's limpid prose. Please, Boaz— finish your book, or start using Sipser's until you're done.
Boaz really cares about his students, but his efforts to to teach theoretical cs in his own way have made the class difficult and confusing. Focusing on more established methods might make the class more enjoyable for all.

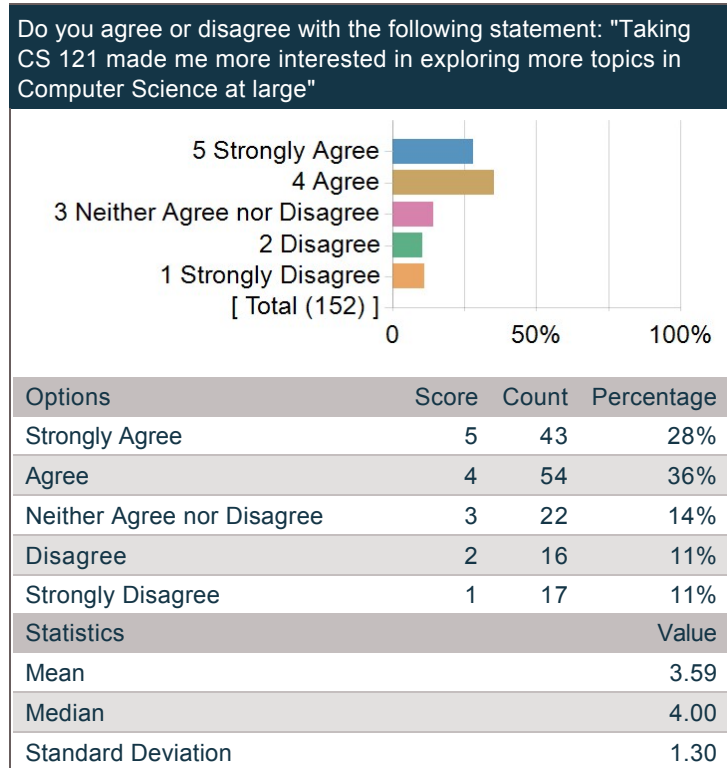
Comments
Professor Barak is an amazing person who loves what he is teaching. His lectures can feel like a lot of information at once because the material is confusing but he truly cares about everyone.
Boaz generates a lot of enthusiasm about the theory of CS. Sometimes lectures can be a little bit confusing, but he is always happy to help understand the topics.
Boaz cares a lot about students' learning and well-being. His personality (and jokes!) help generate enthusiasm for the topics. I also appreciated his inclusion of real-world applications of concepts.

Custom Questions

Do you agree or disagree with the following statement: "Taking CS 121 made me more interested in exploring more topics in Theoretical Computer Science"



Do you agree or disagree with the following statement: "Taking CS 121 made me more interested in exploring more topics in Computer Science at large"



CS 121 is currently required for the computer science concentration. What are your thoughts on this, and in particular having taken the course, do you feel that the concepts you learned are important for your education as a computer scientist?

Comments
I do not think this course is necessary.
Yes!
I feel like the concepts learned were important in giving context for how computing works in theory.
I think it is a good course to take once. It gives you the basics of computation.
No. Not even in the slightest. The fact that this course is so widely considered a nightmare class, that kids will willingly enroll in MIT alternatives to avoid it, and its reputation precedes it as a wholly negative experience, should convey to the CS department that it should not be required.
I think that this course and content was very important for my development as a computer scientist. I consider myself most likely on the CS grad school path and having this foundational material in theory is needed for what I am interested in doing. Even for those going strait to SWE I still think that some sort of theory requirement is good. I see the CS department as foremost academic in its mission and that a good theory class is needed for well rounded education.
Perhaps? I imagine this is very useful for people going into academia but I'm not sure how useful it is for people entering industry. However, I really enjoyed the crypto classes and (finally) understanding what P = NP is and the implications of it.
It should be required. It is important in my education as a computer scientist.
If I chose to pursue theoretical computer science, this would be very important. But for software engineering, I think it would be better to have it as a technical elective.
I think that CS121 should be a required course. While most of its material may not be directly applicable to applied computer science, it is certainly valuable to understand the theoretical foundations of the field, key results, and open problems in computer science theory.
Bring back Harry Lewis! Boaz is an incompetent teacher.
The way this class goes into depth makes it more than an intro. I feel that a lower level intro theory class should be required, but this class went too far for those uninterested in or undedicated to the subject matter
The concepts are important but there is absolutely no need to make it as hard as it is. An easier theoretical computation class should definitely be required (that can still cover the same material but in less depth and with less proofs or less difficult proofs)

Comments
and then a cs121 class should be an optional class taught as a followup if you decide to learn more about theoretical computation. I, for one, know I don't want to study theoretical computation so taking such a hard theoretical class makes no sense for me as I could have spent my time taking a difficult but much more relevant class to my interests.
I despise this. Honestly, the concepts are important! But they're all taught in CS124! I can see the argument that this class should be a preliminary class that people take before 124, to give experience with proof-based math. I can see strongly encouraging sophomores to take it. But I see no reason at all why I should be forced to take this class as a Senior to graduate when I took 124 freshman year. I didn't learn any new concepts in this course, because they're all taught in 124. Encourage the class, yes, but I would frankly be shocked if there's a single student at Harvard who found that got a lot out of this class after taking CS124. It feels like if Math 21 was REQUIRED for all math concentrators, and seniors had to go back and take it after taking 25 / 55 freshman year. Waste of time, please remove this requirement or let it be filled by a higher-level theoretical CS / algos class.
I think that it is important but I'm also interested in theoretical computer science so I might be a bit biased. It will give people a better understanding about the limitation of computers which I believe is very important.
I am for a theoretical computer science class being required, and I think that many of the topics in CS121 should be taught to all computer scientists. Right now, I don't think CS121 is a good enough course for its requirement to be justified.
The first half of the course builds up the theory of computability, and all sorts of weird systems of computation — NAND-TM, NAND-RAM, etc. The second half breezes over a lot of topics very quickly, with a whole lecture devoted to statistics and another to quantum physics. The class ends up feeling like it's neither deep nor broad, which makes me wonder why it's required — it neither surveys a large number of topics nor teaches any one topic super deeply.
As a prospective Math major (with either a joint/secondary in Computer Science), I believe that CS121 is fairly important as it is one of the fields which I would be more likely to pursue should I eventually decide to venture into computer science. Regardless of one's area of specialization, I feel that CS121 is still a valuable class it is allows one to be aware of many significant concepts in the CS field today (such as unpacking the definition of the $P = NP$ problem).
Yes, I think this is good
No, because I don't know how they connect back to everything, nor do I actually know what I am supposed to know
I think the course is probably still useful for computer scientists to learn, but more coding would make the class feel less detached from the rest of the department.
I think as a liberal arts school CS121 should definitely be required it teaches the foundations of computation and truly is a great course in terms of material
Please remove the requirement.
CS 121 should not be a required course. The topics may be of interest to a mathematician or purely theoretical computer science, but it is very removed from most of software development or anything engineering-related.
People take computer science for very different reasons, and it is a very broad field. Making it optional is the best way to serve the students it matters to best
They are important, but the class needs to be heavily reworked — most of what I learned in CS121 was taught, for example, in CS136 in much more intuitive ways.
I think this is an interesting class, but I don't think it should be required. This class helps contextualize a lot of work done, but it did not improve me as a coder or give me a greater understanding of computer science at large (beyond the theoretical field).
If it is required, I wish there were more alternatives than just one course for the CS theory requirement. If this course was not intuitive or extremely overwhelming to a student, it can potentially dissuade a student from the field of computer science entirely, which shouldn't be the case. I think the concepts are important to understand from a higher level standpoint, but this course made the topics extremely difficult to grasp and increased how intimidating it is to study computer science.
Yes, I believe that the concepts in 121 are very important for any computer scientist. Even for students who don't take any more theoretical CS classes or who don't directly apply the course's material, I think the material is invaluable to be in the field.
It's a must-take.
I don't think this class should be required. I think it's a great addition to a computer science education and I'm glad I know this material, but making it required is unnecessary in my opinion. I would rather have more time to learn about data structures and algorithms instead of cramming that class into one semester.
I support the requirement of CS121 in the concentration. I feel that it's important to understand the theory behind computation and suggest that people who do not support the requirement should go to MIT instead.
I think this largely depends. Some people going into Computer Science are interested solely in applied skills, and they might not find theoretical CS knowledge that interesting or useful, while others would find that it constitutes basic, necessary CS knowledge.
The concepts are incredibly useful, but the course causes unnecessary stress and requires a huge amount of time, I feel as though I could have learned just as much or even more with a restructuring of the course.

Comments
I'm actually not a concentrator, but I think it's a valuable approach to the subject and should continue to be mandatory.
I think they are important, but I don't think I have learned much considering that everything moves at too fast of a pace. I wished I had not chosen CS after this class, and I think that is indicative of the course as a whole.
No
The concepts I learned in this class are crucial to being a computer scientist. I am very glad that I took it.
I feel like they are important
sure but why not require algorithms
While I feel as though I will hardly use any of the skills or concepts I learned in this class for actual work, I understand why we have to have a solid theoretical foundation.
I agree that it should be required!
Yeah, I think it's a good thing that CS 121 is required for all CS concentrators.
I think algorithmic analysis is the big one! I do think the concepts are important for my education as a computer scientist.
I think that it should be required. I'm not sure that the concepts would be useful for everyone who studies computer science, depending on their postgraduate plans, but I do think the way of thinking which the course teaches is important to become a good computer scientist.
I agree that it is a very useful topic, but please organize the class better or better publicize the MIT alternative too.
I think it's useful to think computationally, and it seems to give a good foundation for how to think about computers that has relevance in other topics/subfields of computer science.
I think this is a good requirement. I believe many people would not attempt theoretical CS without this requirement, and I know of at least a few people who realized they enjoyed theoretical CS over programming through this class.
Yes, I'm glad this is a requirement.
I feel that most of the topics covered by the course are important to computer science as a concentration, but narrowing down the curriculum to a few of the most significant ones would have been helpful in understanding the course content more thoroughly
No
I think this course was great and that the professor did a great job. The content was interesting and challenging. However, I think problem sets were significantly too difficult and painful.
Yes, but I think the curriculum, particularly the later half, could be altered slightly to make this more apparent.
Yes
I have no idea how I will every use any of these courses in my future as a computer science concentrator or after I graduate. I despised almost every second of this class. I feel extraordinarily incompetent and have very seriously considered switching out of CS as a result. While I did learn a lot after cramming for the exam, I don't know what it was all for, and I have heard that CS 124 does a better job covering important CS theory while still being practically useful and applicable. Maybe in the future I will look back on this course and find it was important for my education as a computer scientist, but right now, I don't feel that way at all. It was especially brutal to hear that my friends who took the MIT version of CS 121 found it clear, interesting, and even useful, although difficult, especially when I compared that with the criticisms I had heard about CS 121, namely that it was very unclear, uninteresting, and unuseful. Also, MIT doesn't require their version of CS 121 for CS majors, and they seem to do more than fine.
I do not really know where these concepts will be useful in the future, besides theory (not yet, at least). However, I did learn a lot and am thankful that I did. I'm not sure if it should be a requirement, since there are a lot of other important concepts in cs that are useful. Overall, I am glad I took the class and since the class will keep improving, I think in the future it will be even more useful. As such, maybe it is not so bad if it is a requirement, though I also believe that students should have more of a choice as to whether they should take theory. Thanks for a wonderful semester of learning!
For a course that is required for the CS concentration, it does a horrendous job introducing students to theoretical computer science and getting them interested in the field. CS20 does a far better job at creating enthusiasm for the subject, but that was all but ruined because of this course. I originally considered getting a Masters in CS but this course made me think otherwise because of how miserable it was.
I do not think it was important for my education as a computer scientist
I think the concepts are helpful but not absolutely necessary.
I did not find it particularly helpful as I am not interested in theoretical computer science, and I didn't feel like it applied to other topics that much
not really, if anything, cs124 should be required instead of this
I understand why the concepts are important to computer science, just wish the teaching/organization was executed better.
Hard to say.

Comments
i feel sad and wish i had more time to do stuff in college and actually try my best for courses. but the stuff was useful and cool, i just didnt get to engage in it that much and i guess thats just a bigger issue at harvard in general
I really enjoyed taking 121. However, I don't know if it should be required. I think many CS concentrators would not mind not having to take this course.
I think the concepts are super interesting and really important for a foundation in computer science, but the class could really use improvement for a required course.
I feel like only a portion of the class is really necessary and important, like P vs NP, reductions, and the basics of turing machines. Everything else does not seem very useful
I don't think that this should be a class that is required for the CS concentration because it doesn't teach any useful or applicable topics unless you're planning to go into CS theory. If you want to become a software engineer, none of the stuff in this class will ever be of use.
Professor Barak seems to be working very hard to make his course enjoyable and meaningful and I think he's doing a great job. I am personally more interested in the applied aspects of computer science, but taking this course has inspired me to explore more theoretical concepts in the future.
Though the concepts may not be directly applicable in industry, I found the subject material quite interesting and enjoyed the proof-based nature of the course. There were many fascinating results, and it was easy to get bogged down in the mathematical notation, but after seeing past this, I was able to enjoy the course.
I do think the concepts are important for my CS education. However, I really struggled in the course—even as a junior with CS experience and a good network within the CS community. I understand it's a tough class, but I think we can provide even more support for struggling students so that they do not fall through the cracks.
This course in its current form should not be required. I took it in my senior year and have not used any of the material covered in it in any of the other CS classes I've taken. In this form, it is only useful for people already extremely interested in theoretical CS as it teaches almost no generally useful content, instead focusing on domain-specific esoteric math.
Yes. Theory is fundamental.
I agree that it should be a requirement.
Some low-level things were frustrating to me at times. But the big ideas were a lot of fun.
Only the topics at the very end of the course sparked my interest in broader topics in computer science. Although this course may be very foundational, I think the department should consider altering the requirement so students can choose between 121 and 124.
I strongly agree with this requirement, since individual programming languages don't last and/or there are just lots of different ones being used in industry, and a theoretical knowledge, while definitely not everything is very important. I definitely think it was important, although I do happen to be biased since I happen to really like TCS, however, there's a reason I like it! :)
Probably?
I think so, but I'm not too sure how. I feel like a better thinker/problem solver though
I don't think it is necessary to be a better programmer or for software engineering but it helped me understand what the work of a theory computer scientist work entails and the big ideas / fundamental problems (P vs. NP, uncomputability) in computer science.
Seems fine that it is required, but the ambiguity in rigor of the class as well as some seemingly unrelated unimportant parts of the class have me confused.
I think this class should not be a requirement.
I feel like it is a very large step between CS51/CS50 and CS121. I think there should be other options or this course should be decreased in difficulty.
No
I feel that this has been important for my education as a computer scientist.
I support CS 121 being a required course for CS concentrators. Although the course material may be rather difficult for some, especially those who don't have much prior background in proof-based mathematics, I believe the topics discussed in the course are necessary to gain a well-versed understanding of computer science beyond just programming and tech internships.
It was useful but would easily have made me switched concentrations if I had not been too far in.
I don't think CS 121 should be required for the concentration, or at least if it remains required, we should be communicated to better about WHY it is important for us to learn theoretical CS.
If I intended to continue in academic computer science, perhaps. However, the mathematically-focused approach was frustrating at times.
I think CS121 was helpful in building my basic general knowledge of the field and helping to advance my intuition at the discrete math which is crucial to CS.
I guess I would probably take it even if it wasn't required, but it wasn't my favorite course.

Comments
Probably not in the grand scheme of things, at least this is what people working in the tech industry tell me. Of course, preparing you for industry is not the point of a CS education at Harvard. However, having friends that go to institutions like Georgia Tech, doesn't really seem to be a necessary part of my undergraduate experience either.
I think learning the math behind the fundamentals in computer science is really important for making you understand where the field is today and where it came from.
I think it is a great way to align and prepare students for technical concepts and rigorous proofs. Definitely support having CS121 as a requirement.
Yes. This course should absolutely be required for the CS concentration. This is vital information for a computer scientist to know.
This class should absolutely not be required. I am a senior taking the course, so I've taken most of my CS requirements already and think I have a pretty good perspective on this. I think many aspects of this course scare students away from CS, and the course is not taught/organized well enough to excite students about the material (which I do believe could be interesting if the class wasn't as frustrating). A class like Cs124 which is better organized, better taught, has a mix of theoretical and practical material, and covers a wider range of things would I think be a better required course
I agree, it should be required. This information is crucial to learning all subdomains of CS and is necessary to advance how we understand computation.
This does not feel very applicable for my future as a CS degree holder given that it was so difficult, but I did learn a lot and enjoyed my time, even though my GPA will definitely suffer disproportionately to how much time I invested in the class.
I think CS 121 should definitely be required for CS concentrators. On the surface, it may seem like the content covered in CS 121 is too theoretical to be applicable to the education of most CS concentrators. However, I firmly believe that the topics we covered in this class are crucial in order to gain a more fundamental understanding of the modern developments in computation, why certain computational models exist, and what we can gain from them. This is especially true since the implications of topics such as the P vs. NP conjecture are evidently tremendous. It may be difficult for students to appreciate the course while taking it, especially if they find it challenging, but in the long run, I think it is very important to take this class to be able to see the bigger picture in the field of CS.
I think it makes sense to have this as a requirement, the concepts are pretty fundamental to computer science at large.
I am somewhat glad CS 121 was required, because I probably wouldn't have taken it otherwise. I personally really enjoyed the class and am grateful to have taken it. But some of the material we learned doesn't seem like it would be particularly important for our overall education as computer scientists? I suppose it depends on our individual interests and what we end up doing in the future. But for me, personally, I am glad that I was required to take this class.
Although I don't think I will use what I learned in this class at any point in the future, I think it's a good thing that it is required. I think it's good for CS students to get a wider view of what is going on in the field than just useful classes would give them.
Honestly, not really. Maybe generally having a mind towards time/space efficiency in your programs is useful, but it's not like CS121 where you're taught how to implement these efficient algorithms.
Yep! You need to know whether a problem is solvable before even solving it
I do not believe it should be required for the CS concentration, as it is not necessary for 124.
I think it is important information to know as a computer scientist. I can't imagine calling myself a student of CS if I didn't have a firm grasp at theoretical CS
I found it a little hard to see how some of the topics are more broadly related to CS applications, as someone who is more interested in the application side of things
I don't love the material, but I can see why it would be a CS requirement, just as I can see why 124 would be a requirement.
This is quite a controversial question. I personally found CS121 really interesting and Boaz made me really like the class. Without Boaz, however, I am inclined to believe that the experience would have been painful and perhaps not worth it. To be honest, I don't know enough about computer science to say whether or not this will be important for my education but I found the material interesting for sure!
I am glad that it is required because if it was not, I would not have taken a theoretical computer science course. After taking the course, however, I gained an appreciation for theoretical CS and have come to see how foundation and important it is to know.
I think this course is a good complement to some of the more applied courses.
This course is absolutely important in separating the computer science concentration from just being a programming concentration. Valuable and interesting course, which hopefully exposes people in the computer science department to mathematical rigor they might not be used to.
I feel ambivalent. On one hand, I probably would not have taken this class if it wasn't required, but looking back I'm kinda glad I took it because I do think some of the concepts taught are cool. That said, I don't think it should be a hard requirement for every CS student, maybe the requirement should be for everyone to take 2 theory courses but not necessarily make 121 one of them.
I think theoretical computer science and knowledge of complexity classes is important for analyzing algorithms in the real world, and thinking about computational models like Turing machines and circuits are important for knowing what computers can and

Comments
cannot do. Otherwise, the topics are pretty interesting on their own so even they were not useful, the course would still be worth taking.
Yes. CS121 is the best introduction that one can get to what a CS degree means i.e it's more than just getting a tech job or software development. Becoming a CS scientist seems cool after seeing how logical and challenging the problems that CS scientists are working on are.
I'm not convinced that this course was a necessity for my computer science education. I think it should be required for students who want to pursue computer science more academically, but in the words of Boaz himself, I did not find the course "directly useful" to, for example, software engineering/development (which is what I'll likely end up doing).
I feel the fundamental theory in 121 is absolutely necessary for CS students. It really inspired me to look at more classes in TCS and possible research opportunities as well.
not really, a lot of it was covered in cs124 and i'm not really that interested in theory. Don't think I'll really use much of this material in the future.
I do agree that the course concepts are important for the computer science concentration. That being said, I think that the course itself is wholly terrible, and that starts at the top. Boaz might be the worst instructor I have encountered at Harvard. I have said this countless times in this course review, but using his textbook for this class is completely indefensible. Harvard CS needs to seriously review both this text, and Boaz's teaching credentials. As long as Boaz is leading this course, it should not be required for CS concentrators.
I agree that this is an important course
I think this course should continue to be required. The course is not only practical for people who want to go into software engineering, but also opens doors for people who had no idea what theoretical computer science was.
Yes
The concepts are no doubt important to an understanding of computer science, so this course should probably be mandatory. However, it's a shame that this is the case because the course is very poorly taught.
There should be some sort of theoretical requirement for the cs concentration, but this version of 121 should not be the class to fulfill that requirement. Some of the concepts are important, but there is a better way to teach them.
I think this makes sense because it is a very different path in computer science than the programming classes that everyone thinks of when they choose CS. If CS121 wasn't mandatory I would not have taken it, and would not have gained an appreciation for theoretical cs. I actually found the material interesting and it is something I would consider studying further, if I didn't find the class so difficult
As a computer scientist, definitely, but maybe not as much as a programmer or engineer.
Yes.
I think it should stay as a required course. The concepts I learned are important for my education as a computer scientist. Furthermore, because the course can be challenging some students might want to skip it, but I think it is really important so making it mandatory is good.
CS 121 should definitely remain a requirement for CS concentrators. Understanding concepts like computability, P vs NP, and the efficiency of algorithms is critical.